【新たなテクノロジーとCBDC】
ワーキンググループ（WG４）について

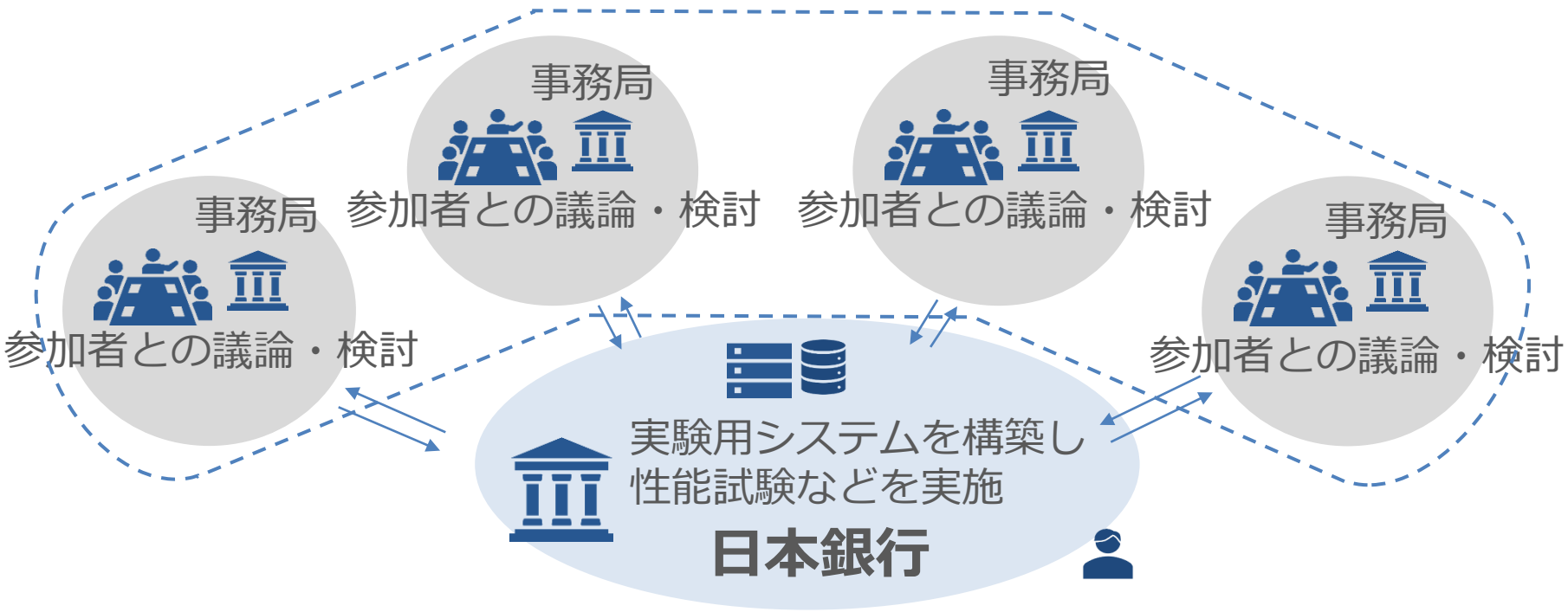
2024年1月

日本銀行 決済機構局

WG4の概要

CBDCフォーラムの位置づけ

- パイロット実験では、①エンドツーエンドでの処理フローの確認等のため、**日本銀行が実験用システムを構築し、性能試験等**を行う（**実験用システムの構築と検証**）とともに、②CBDCの制度設計を適切に進める観点から「**CBDCフォーラム**」を設置し、**リテール決済に関わる民間事業者の参加**を得ながら、**幅広いテーマを議論・検討**することとしている。
- ①・②の検討成果は、必要に応じてお互いの作業にフィードバックすることを想定。



WGの組成と議論・検討テーマ（現時点のもの）

WG名		検討テーマ
[WG1] 9/20日～	CBDCシステムと 外部インフラ・システム等との接続	勘定系システムとの接続
		民間決済インフラとの接続
		既存のインターネットバンキングアプリ等との連携
[WG2] 9/26日～	追加サービスと CBDCエコシステム	CBDC のビジネス活用（追加サービスのあり方）
		追加サービスにかかるCBDC システムの外部連携
		CBDC エコシステムのデザイン
[WG3] 10/25日～	KYCとユーザー認証・認可	KYC、AML/CFT の実施
		認証・認可
[WG4] 1/30日～	新たなテクノロジーとCBDC	バックエンド（代替的な台帳データモデル等）
		フロントエンド（「ウォレット」等）
		他の決済手段や資産との共存（ステーブルコイン、アセットトークナイゼーション、DLT基盤との相互運用性等）
	ユーザーデバイスとUI/UX	UI/UX、アクセシビリティ
		エンドポイントデバイス
		オフライン決済
	他の決済手段との水平的共存	電子マネー等との交換容易性
	基本機能の事務フロー	基本的な機能にかかる事務フロー
		現金とCBDCの交換

WG4参加者一覧

- コインチェック株式会社
- セコム株式会社
- ソラミツ株式会社
- 大和証券株式会社
- 株式会社日本証券クリアリング機構
- 野村證券株式会社
- 株式会社三井住友銀行
- 三井住友信託銀行株式会社
- 株式会社メルペイ
- 株式会社BOOSTRY
- 株式会社Datachain
- 株式会社JPX総研
- 株式会社NTTデータ
- SBI R3 Japan株式会社
- 株式会社Startale Labs Japan

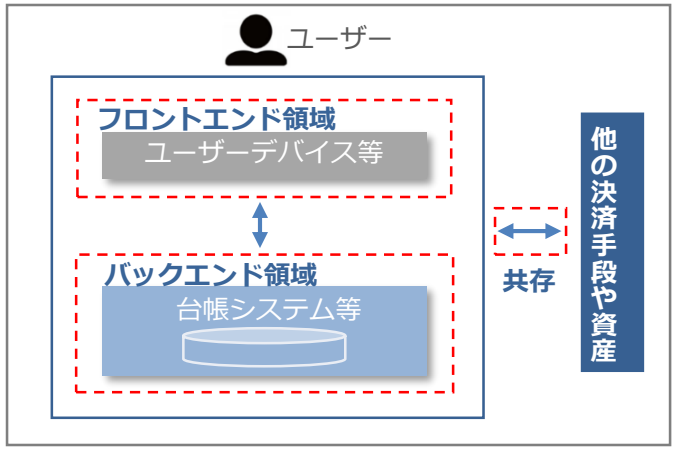
（五十音・アルファベット順）

WG4の進め方

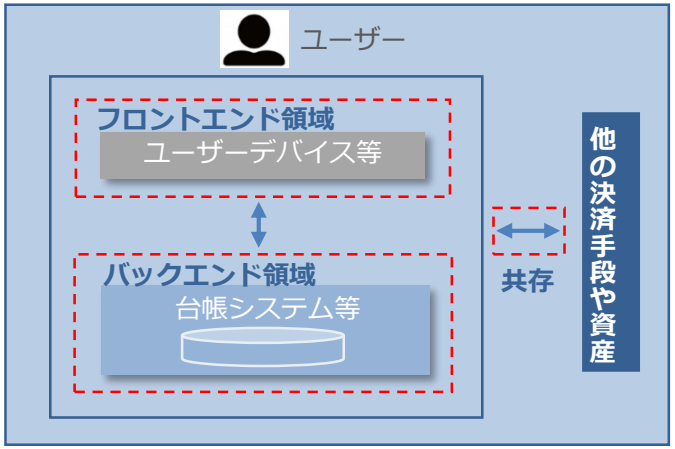
- **WG4「新たなテクノロジーとCBDC」**は、CBDCシステムに関連する新たな技術に関する理解を深め、活用可能性などを検討。
- 参加者からのプレゼンテーションなどを踏まえ、新たな技術から着想を得る形で**CBDCへのインプリケーション**などについて理解を深めていく。

バックエンド領域 → **他の決済手段や資産との共存** → フロントエンド領域

【WG4以外】現状整理から展望



【WG4】新たなテクノロジーから着想



各領域での論点の例

バックエンド領域

バックエンド領域から議論を始める

- 台帳システム
- その他のシステム・ネットワーク

他の決済手段や資産との共存

- ステーブルコインとの関係
- アセットトークナイゼーション（セキュリティトークン等）との関係
- DLT基盤との相互運用性
- その他

フロントエンド領域

- エンドポイントデバイス
  - ✓ いわゆる「ウォレット」のあり方
- その他のシステム・ネットワーク

バックエンド領域の検討の進め方

バックエンド領域の検討の進め方

- 日本銀行の概念実証では、トランザクション集中時のパフォーマンス低下への対応や、スケールアウトといった台帳データベースの性能拡張方法について、検討を深めていく必要性が示唆された。
- バックエンド領域の検討の進め方としては、まずは台帳システムについて、トランザクションが集中するなかで、高スループット/低レイテンシといった高い性能を実現するための対応策について検討したい。

バックエンド領域

- 台帳システム
  - ✓ 性能面の論点（スループット/レイテンシ等）
  - ✓ その他の論点
- その他のシステム・ネットワーク

トランザクションが集中するなかで、高い性能を実現するための対応例

・台帳のデータモデルに関する工夫
・データ更新時のロック範囲の工夫
・データ更新の仕方の工夫
・DB機能や実装上の工夫

最初の数回はこれらのテーマを扱う

性能向上のためのアプローチ

- 概念実証でも確認した通り、トランザクションが集中する中で、高い性能を実現するためには、様々なアプローチが考えられる。
  - 台帳のデータモデルに関する工夫 ← 今回はこのテーマを扱う
    - ✓ 「口座残高型」以外のデータモデルの採用
  - データ更新時のロック範囲の工夫
    - ✓ 口座やレコードの分割
      （例：トランザクションが集中する口座にサブ口座や入金専用口座を設定）
  - データ更新の仕方の工夫
    - ✓ 追記のみの更新を行う
  - DB機能や実装上の工夫
    - ✓ DBのシャーディング
    - ✓ インメモリDB
    - ✓ いわゆるNo-SQLやNew-SQL DBの活用
- 複数のアプローチを組み合わせることも可能と考えられる。

留意点

- 前述のような性能向上策をとった場合の影響は考える必要がある。
  - ✓ 実装や運用の難度が上がる可能性があるか？
  - ✓ 処理の順序性への追加的な配慮が必要となる可能性があるか？
  - ✓ 冗長性や整合性への追加的な配慮が必要となる可能性があるか？

- また、性能面以外の観点からも、技術の特徴をみていく必要がある。
  - ✓ プライバシー保護が容易になる？／難しくなる？
  - ✓ プログラマビリティを実現しやすくなる？／難しくなる？
  - ✓ 外部システムと連携しやすくなる？／難しくなる？

- 上記の観点も踏まえて、技術的選択の可能性を考えることが重要。

台帳のデータモデルについて

台帳のデータモデルの分類

- 台帳の性能向上のためには多くのアプローチがあるなかで、まずはデータモデルに関する検討から始める。
- データモデルの分類については、例えば以下のマッピングがありうる。

	口座残高型	口座残高型以外
状態を更新	● 金融機関の勘定系システム ● 日本銀行の概念実証（口座型）	● 米Project Hamiltonフェーズ1（UTXO） ● デジタルユーロ・プロトタイピング（UTXO） ● 日本銀行の概念実証（トークン型）
状態の変化を蓄積	● イーサリアム	● ビットコイン（UTXO）

（※）日本銀行の概念実証において検討対象としたデータモデルは、以下の通り。
・口座型：CBDCの保有状況をユーザが有する口座の残高として認識するモデル
・トークン型：価値が付された金額データに固有の識別子（ID）を付与し、そのIDとユーザIDの紐づけによりCBDCの保有状況を認識するモデル
なお、台帳システムの設計パターンは、データモデルだけではなく、台帳の管理の仕方も考慮した4種類を検討対象とした（詳細は参考資料を参照）。

分類の軸①：「口座残高型」か「口座残高型以外」か

- 口座残高型の場合

現在の状態

利用者	残高
A	50
B	70
C	30

トランザクション (Tx)

操作
Aの残高を45減らす Bの残高を45増やす

(AがBに45円を送金)

Tx実行後の状態

利用者	残高
A	5
B	115
C	30

- 口座残高型以外、例えばUTXOの場合（※後述するProject Hamiltonでの考え方）

「システム内で未だ使用されていない」 Unspent Transaction Output=「現在使用可能な」価値情報
全UTXOを記録・更新するデータベース (UTXO set) で、デジタル通貨の流通を表現

現在の状態
(UTXO set)

価値	所有者
10	A
20	B
30	C
40	A
50	B

トランザクション (Tx)

費消 (入力)		追加 (出力)	
10	A	45	B
40	A	5	A

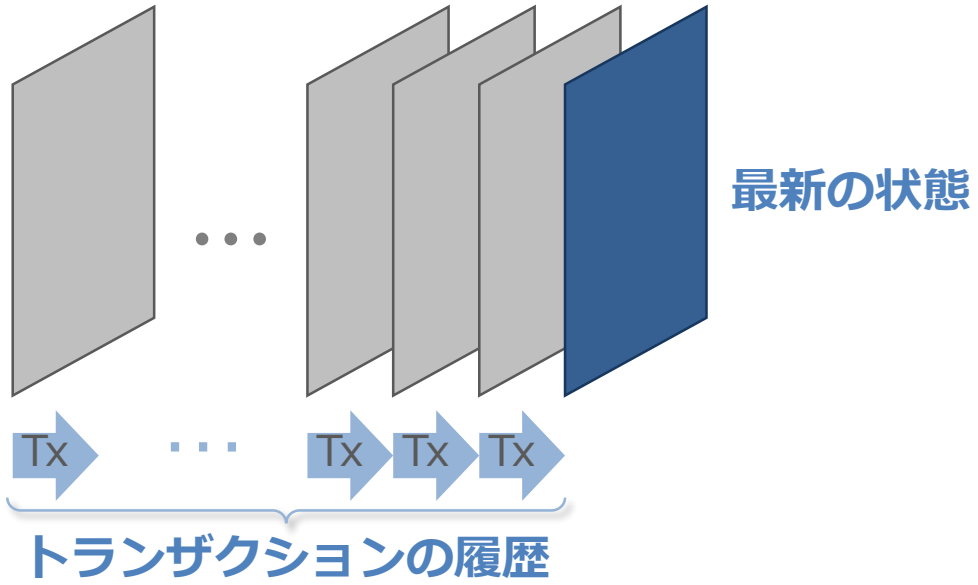
(AがBに45円を送金)

Tx実行後の状態

価値	所有者
20	B
30	C
50	B
45	B
5	A

分類の軸②：「状態を更新」するか「状態の変化を蓄積」するか

- 伝統的な金融機関の勘定系システムのデータモデルでは、トランザクションの最新の状態を管理することを重視。
- 他方でイーサリアムやビットコインのデータモデルは、トランザクションの履歴を、ブロックチェーンで記録することを重視。

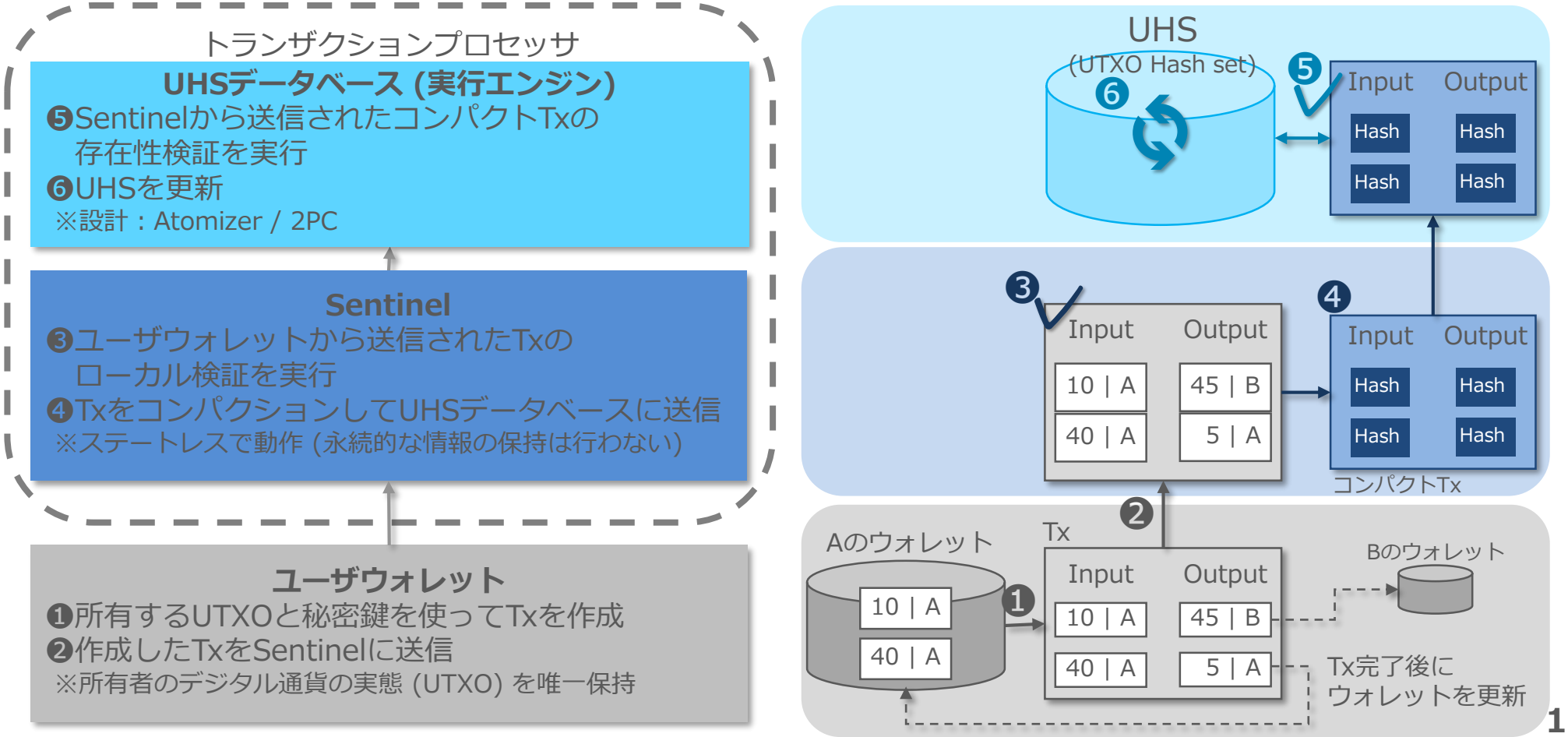


「口座残高型以外」×「状態を更新」の事例の紹介

- 以下では、「口座残高型以外」×「状態を更新」の事例をご紹介。
  - ✓ 事例①：Project Hamilton
  - ✓ 事例②：デジタルユーロ・プロトタイピング
  - ✓ 事例③：日本銀行の概念実証

事例の紹介①：Project Hamilton

- ボストン連銀とMITメディアラボデジタル通貨イニシアティブが共同でProject Hamiltonを実施、2022年2月にフェーズ1報告書を公表。
- 実験システムの価値情報はUTXOモデルで表現。ローカル検証をするSentinelと、存在性検証をするUHSデータベースで構成。UHSデータベースの設計はAtomizerと2PCの２種類（詳細は参考資料を参照）。



事例の紹介①：Project Hamilton

- Atomizer: シャード数8でピーク170,000TPSを観測。
- 2PC: シャード数に対して線形にスケールし1,700,000TPSを観測。

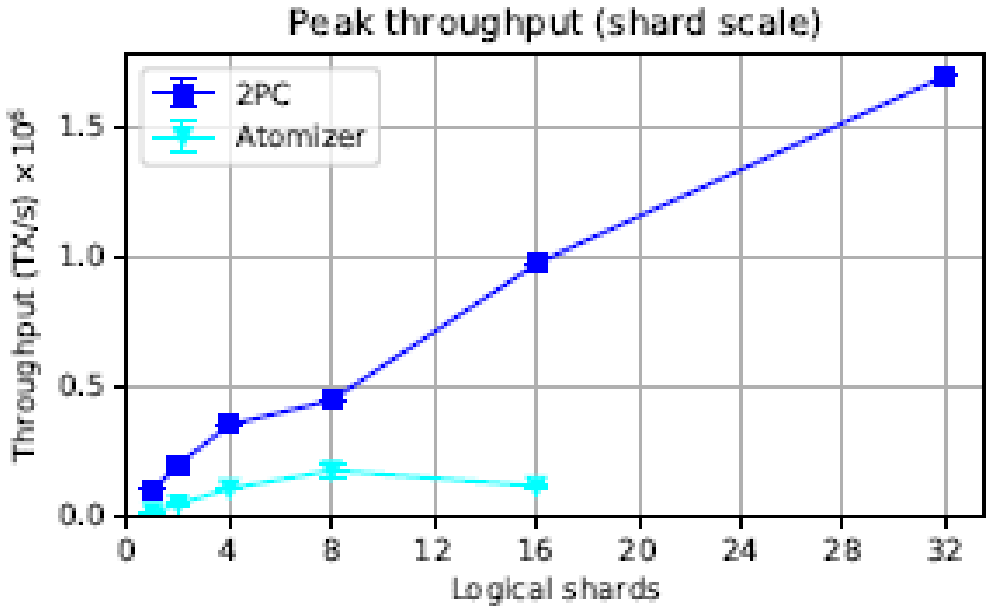


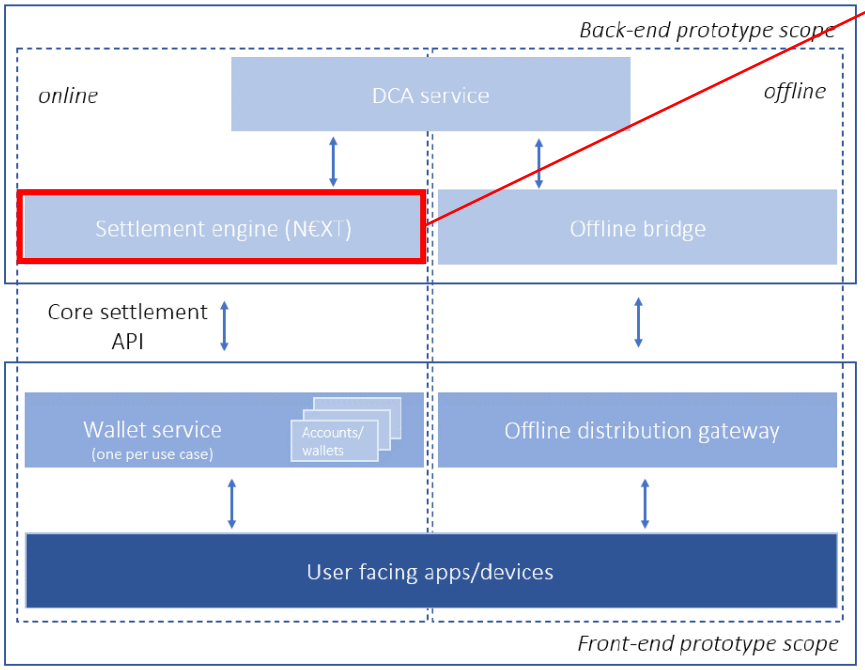
Figure 10: Peak throughput of the atomizer and 2PC architectures when varying logical shard count to be 1, 2, 4, . . . , 32.

事例の紹介②：デジタルユーロ・プロトタイピング

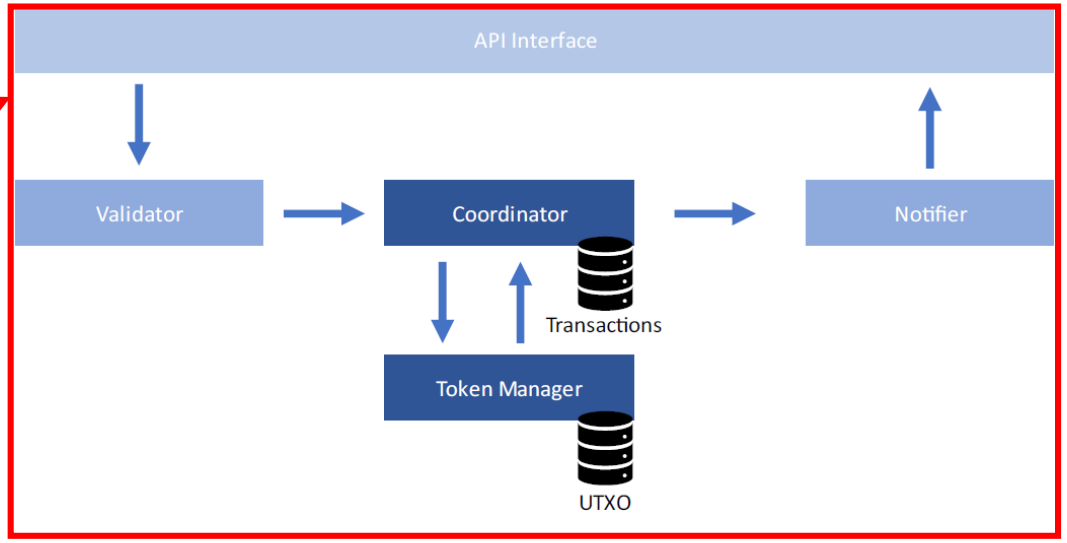
- ECBもUTXOモデルに注目。
- デジタルユーロの調査フェーズの一環で2022年7月～23年2月にプロトタイピング演習を実施。プロトタイピングにおける決済エンジンN€XTの価値情報はUTXOモデルで表現され、シャーディングされた中央データベースで管理。
- アーキテクチャや性能に関する詳細な情報は公表無し。

【プロトタイプの各領域】

Figure 1
Front- and back-end prototype scope



【N€XTのアーキテクチャ】



Validator

・Txのローカル検証

Coordinator

・Txの存在性検証
・2PCの司令塔としてシャーディングされたDBの更新指示

Token Manager

・UTXOを保持
・最新の状態に更新

留意点

- Project Hamiltonは、決済のコアエンジンであるトランザクションプロセッサを実験対象としており、測定された性能値は、end-to-endのシステムで見たものではない。
- Project Hamilton、デジタルユーロ・プロトタイピング共に、今回紹介したアーキテクチャは、あくまで実験の特定のフェーズで用いたものにすぎない。
  - Hamiltonも、後続のフェーズ2では口座残高型（ERC20）のアーキテクチャを採用。
    — 汎用のプログラマビリティを備えるスマートコントラクトの実行をサポートしながら、118,000TPSのスループットを観測。

事例の紹介③：日本銀行の概念実証

- 日本銀行の概念実証では、口座残高型とそれ以外のデータモデルについて、どちらも検討。
- 口座残高型に関しては、同一口座へのトランザクション集中に伴う処理遅延（レコードロックの影響）への対応が必要であることを確認。対応策の一つとして台帳に記録される口座残高データを複数に分けるレコード分割を検討。
- 口座残高型以外のデータモデルとしては、例えば、UTXOモデルに類似したトークンモデルについても検討。

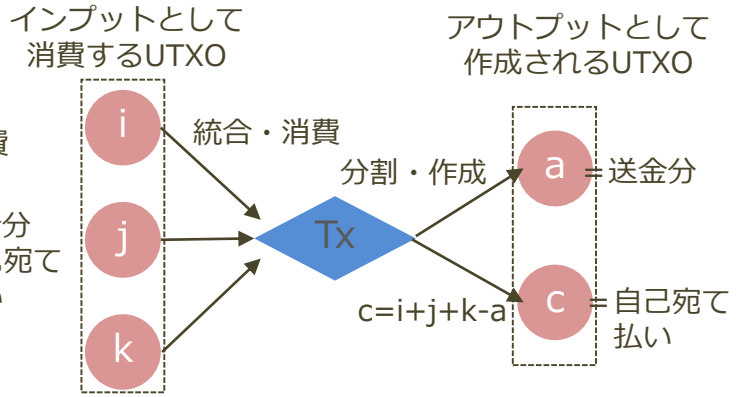
例）Aが100円分のトークンを用いBに60円送金

トークンID	金額	保有者ID
T001	50	A
T002	50	A
T006	500	B
:	:	:

⇒

トークンID	金額	保有者ID	
~~T001~~	~~50~~	~~A~~	＝消費
~~T002~~	~~50~~	~~A~~	
T006	500	B	
T008	60	B	＝送金分
T009	40	A	＝自己宛て払い
:	:	:	

送金トランザクションイメージ



- 口座残高型と比べると、複数のリクエストを並列に処理できるトークンの特徴が確認された一方で、リソース消費量が増える可能性や全体的な実装難度が上がる可能性も確認。

今後のスケジュール

当面の開催スケジュール

- 現時点で確定している日程は、以下のとおり。

	開催予定日	テーマ		プレゼンター
第1回	1月30日 14:00～16:30 <対面とオンライン>	概要	スコープ、事例紹介	日本銀行
		バックエンド領域	UTXOモデルの特徴	SBI R3 Japan 株式会社
第2回	3月13日 14:00～16:30 <オンライン>	バックエンド領域	ご相談中	SBI R3 Japan 株式会社
		バックエンド領域	ご相談中	コインチェック 株式会社
第3回	4月10日 14:00～16:30 <オンライン>	現在調整中		
第4回	5月14日 14:00～16:30 <対面とオンライン>	現在調整中		

参考資料

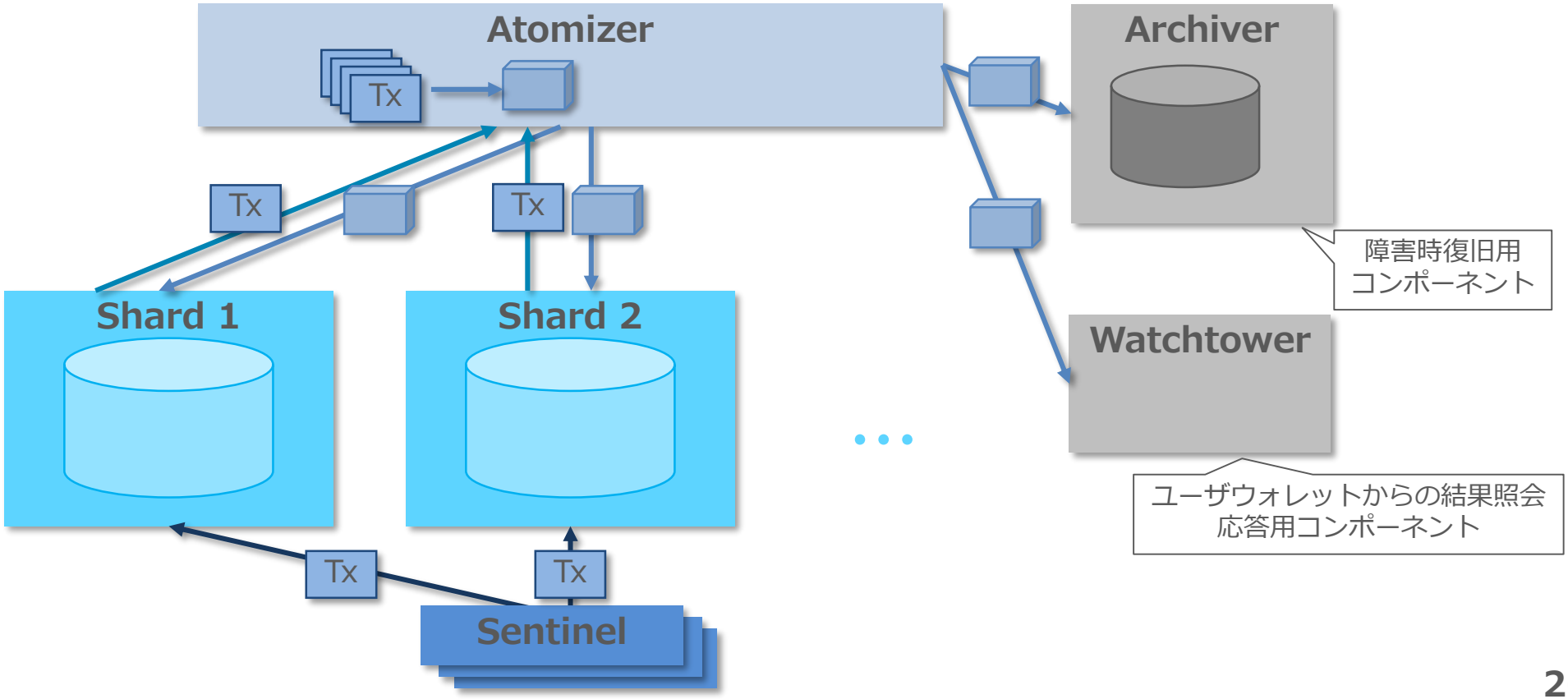
日本銀行の概念実証における台帳設計パターン

- 概念実証で検証対象とした台帳の設計パターンは、①データモデルに関して「口座型かトークン型か」、②台帳管理に関して「中央銀行による中央管理か、中央銀行と仲介機関での分担管理か」の組み合わせについて、以下の４種類とした。

	中央管理	分担管理
口座型	パターン1	パターン2
トークン型	パターン3	パターン4

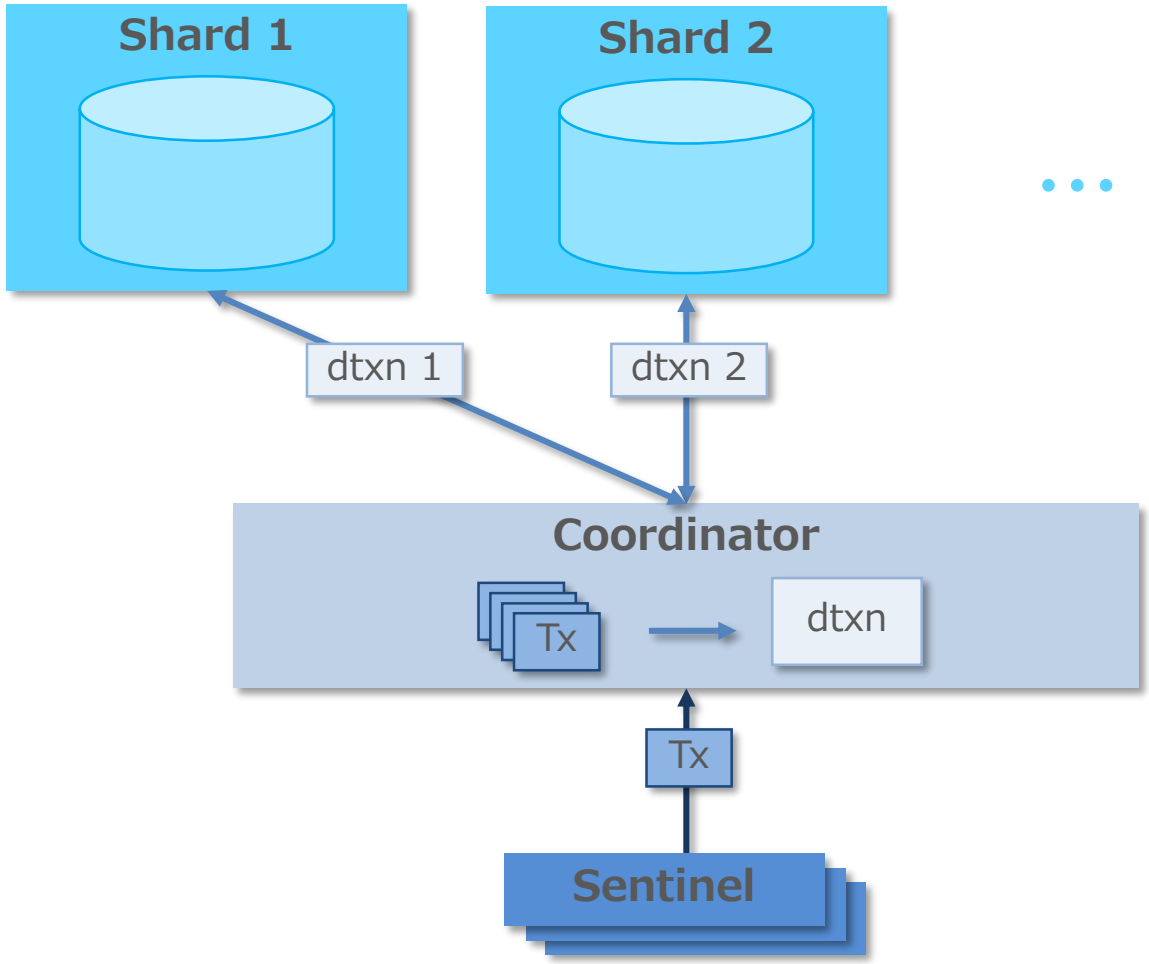
Project Hamilton：Atomizerアーキテクチャ

- トランザクションのAtomicity (不可分性、原子性) を保証する管理コンポーネント Atomizer を中心に据えたアーキテクチャ。
- トランザクションは、分散されたUHSデータベースの各シャードで、まず存在性検証が行われる。その後Atomizerに収集され、線型に順序 (全順序) 付けられたうえ、マイクロバッチ処理で全シャードを更新。



Project Hamilton：2PCアーキテクチャ

- 分散トランザクションアルゴリズム2-Phase Commit (2PC, 二相コミット) の亜種に基づくアーキテクチャ
- トランザクションは、先ずCoordinatorに送付され、2PCベースの分散トランザクション (dtxn) に再構成（複数Txをまとめたマイクロバッチでコミット）



*"**A one size fits all database doesn't fit anyone**"*
(Werner Vogels, CTO and VP of Amazon)

*"**An idea is nothing more or less than a new combination of old elements**" (James Webb Yong, Head of J.Walter Thompson)*