(19)日本国特許庁(JP)　(12)公表特許公報(A)　(11)公表番号
特表2025-520050
(P2025-520050A)
(43)公表日　令和7年7月1日(2025.7.1)

| (51)国際特許分類 | FI | テーマコード（参考） |
|---|---|---|
| *G06F 9/44 (2018.01)* | G06F 9/44 | 5B376 |

審査請求　未請求　予備審査請求　未請求　（全38頁）

(21)出願番号　特願2024-569061(P2024-569061)
(86)(22)出願日　令和5年5月19日(2023.5.19)
(85)翻訳文提出日　令和6年11月28日(2024.11.28)
(86)国際出願番号　PCT/CA2023/050696
(87)国際公開番号　WO2023/225744
(87)国際公開日　令和5年11月30日(2023.11.30)
(31)優先権主張番号　17/664,545
(32)優先日　令和4年5月23日(2022.5.23)
(33)優先権主張国・地域又は機関
米国(US)
(81)指定国・地域　AP(BW,CV,GH,GM,KE,LR,LS,MW,MZ,NA,RW,SD,SL,ST,SZ,TZ,UG,ZM,ZW),EA(AM,AZ,BY,KG,KZ,RU,TJ,TM),EP(AL,AT,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR,GB,GR,HR,HU,IE,IS,IT,LT,LU,LV
最終頁に続く

(71)出願人　321005788
ケーエヌエヌエックス コーポレーション
カナダ国、オンタリオ州 エム５ジェイ ２エイチ７、トロント 、５５ ユニバーシティ アベニュー、スイート １２００
(74)代理人　100104411
弁理士　矢口 太郎
(72)発明者　スリバスタバ、ニーラジ
カナダ国、オンタリオ州 エム５ジェイ ２エイチ７、トロント、５５ ユニバーシティ アベニュー、スイート １２００

Ｆターム（参考）　5B376　BC31 FA15 FA17 FA21

(54)【発明の名称】　ＡＳＩＫ：ブロックチェーンへのモジュール式インターフェイス

(57)【要約】
【要約】 ＡＳＩＫ（Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｋｅｒｎｅｌ：特定用途向け統合カーネル）とは、開発者がブロックチェーンを理解することなく、主要なビジネスニーズに対する汎用的なソリューションを提供するモジュール式インターフェースのことである。前記ＡＳＩＫは、ブロックチェーン上で頻繁に実行される重要なユースケースのためのＡＰＩの集合体である。前記ＡＰＩはチェーンコード上のビジネスロジックと同期している。前記ＡＳＩＫのウェブサービス層はビジネス要件を特定し、タスクや操作を実行するために必要となる可能性のあるバンドルから表現的状態遷移（ＲＥＳＴ）ＡＰＩを公開する。エンドユーザーによって行われた特定のトランザクション要求は、前記要求から収集された入力データによって構成され、チェーンコードによってブロックチェーントランザクションに変換される、あらかじめ想定されたシナリオの一般的な集合体によって実行される。前記トランザクションがブロックチェーン上で成功裏に実行されると、レスポンスは実行可能なアウトプットの形でユーザ

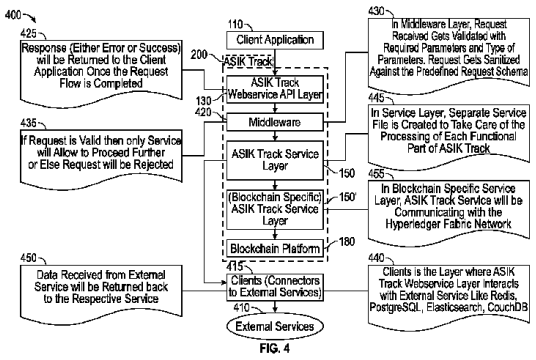


FIG. 4

【特許請求の範囲】
【請求項１】
ブロックチェーンを管理するブロックチェーンプラットフォームにクライアントアプリケーションをインターフェースするアプリケーション固有の統合カーネルであって、
前記クライアントアプリケーションから前記ブロックチェーン上でトランザクションを実行するためのリクエストを受け取るアプリケーションプログラミングインターフェース（ＡＰＩ）層であって、前記ＡＰＩ層は、前記リクエストに関連するデータをフォーマットする、前記アプリケーションプログラミングインターフェース（ＡＰＩ）層と、
前記リクエストに関連付けられた少なくとも１つのコマンドとフォーマットされたデータを受け取り、前記リクエストのタイプを識別し、前記リクエストの前記タイプに基づいて、前記ブロックチェーン上で実行するために前記少なくとも１つのコマンドとフォーマットされたデータを転送し、前記ブロックチェーン上の前記少なくとも１つのコマンドとフォーマットされたデータの処理結果を前記クライアントアプリケーションに返すサービス層であって、前記サービス層は、さらに、前記アプリケーション固有の統合カーネルの所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用する事前設定された特徴と機能を記述する、前記サービス層と、
前記所定のユースケースのための前記事前設定された特徴と機能に従って、前記少なくとも１つのコマンドを前記ブロックチェーンプラットフォーム上で実行可能な少なくとも１つのトランザクションに変換し、前記少なくとも１つのトランザクションの実行結果および前記ブロックチェーンプラットフォーム上のフォーマットされたデータを前記サービス層に返すことによって、前記所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用する前記事前設定された特徴と機能に対するデータ操作を管理するチェーンコードと
を有する、アプリケーション固有の統合カーネル。
【請求項２】
請求項１記載のアプリケーション固有の統合カーネルにおいて、前記ＡＰＩ層は、前記クライアントアプリケーションが、前記アプリケーション固有の統合カーネルの前記所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように前記事前設定された特徴と機能にアクセスするために使用するＡＰＩを公開する、アプリケーション固有の統合カーネル。
【請求項３】
請求項１記載のアプリケーション固有の統合カーネルにおいて、前記ＡＰＩ層は、負荷分散された表現状態転送（ＲＥＳＴ）ＡＰＩを前記クライアントアプリケーションに提供し、前記ＲＥＳＴ　ＡＰＩは、前記ブロックチェーンプラットフォームと相互作用するように適合される、アプリケーション固有の統合カーネル。
【請求項４】
請求項１記載のアプリケーション固有の統合カーネルにおいて、前記ＡＰＩ層、サービス層、およびチェーンコードは、複数のクライアントアプリケーションが、前記ブロックチェーンプラットフォーム上でデータを作成および管理できるマルチテナントフレームワークをサポートするように適合されている、アプリケーション固有の統合カーネル。
【請求項５】
請求項１記載のアプリケーション固有の統合カーネルにおいて、前記ＡＰＩ層は、前記クライアントアプリケーションからの前記リクエストを検証し、前記クライアントアプリケーションからの前記リクエストで受け取ったユーザー詳細を検証し、前記クライアントアプリケーションからの前記リクエストを実行するための前記ユーザーの許可をチェックし、前記リクエストを前記ブロックチェーンプラットフォームに提出するために準備し、前記ブロックチェーンプラットフォームから前記クライアントアプリケーションに応答を返す、アプリケーション固有の統合カーネル。
【請求項６】
請求項１記載のアプリケーション固有の統合カーネルにおいて、前記ＡＰＩ層、サービ

ス層、およびチェーンコードは、プラットフォーム管理者、テナント、およびコアＡＳＩＫ参加者を有する３つの役割をサポートするように適合され、前記プラットフォーム管理者は、クライアントアプリケーションの登録を担当し、テナントは、前記フォーマットされたデータに対して作成、読み取り、更新、および削除の操作を実行するために使用できるＩＤと割り当てられたリソースを持ち、前記コアＡＳＩＫ参加者は、前記ＡＰＩ層との要求を開始するために使用される独自のＩＤを持つ、アプリケーション固有の統合カーネル。

【請求項７】

請求項６記載のアプリケーション固有の統合カーネルにおいて、さらに、各テナントが独自のデータベースで操作を行う、各テナント用のデータベースを有するものである、アプリケーション固有の統合カーネル。

【請求項８】

請求項１記載のアプリケーション固有の統合カーネルにおいて、前記サービス層は、セマンティクスおよびシンタックス、ならびに少なくとも１つのコマンドとフォーマットされたデータが、前記アプリケーション固有統合カーネルの前記所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用する少なくとも１つのインターフェースを概説することによって、前記事前設定された特徴と機能を記述する、アプリケーション固有の統合カーネル。

【請求項９】

請求項１記載のアプリケーション固有の統合カーネルにおいて、前記アプリケーション固有の統合カーネルの前記所定のユースケースは、前記ブロックチェーン上で実行されるビジネスサービスに対応する、アプリケーション固有の統合カーネル。

【請求項１０】

請求項１記載のアプリケーション固有の統合カーネルにおいて、前記チェーンコードは、ブロックチェーン台帳と前記ブロックチェーンプラットフォームのスマートコントラクトにアクセスして前記ブロックチェーン台帳を照会または更新する、アプリケーション固有の統合カーネル。

【請求項１１】

請求項１０記載のアプリケーション固有の統合カーネルにおいて、前記クライアントアプリケーションのすべての操作は、前記ブロックチェーン台帳に安全に記録される、アプリケーション固有の統合カーネル。

【請求項１２】

請求項１記載のアプリケーション固有の統合カーネルにおいて、さらに、状態データベースを有するものであり、前記サービス層が、前記ＡＰＩ層からの入力を受け入れ、前記チェーンコードと前記状態データベースを呼び出すことによって入力をアクションに変換するものでる、アプリケーション固有の統合カーネル。

【請求項１３】

ブロックチェーンを管理するブロックチェーンプラットフォームにクライアントアプリケーションをインターフェースする方法であって、

アプリケーションプログラミングインターフェース（ＡＰＩ）層によって、前記ブロックチェーン上でトランザクションを実行するための前記クライアントアプリケーションからのリクエストを受け取る工程と、

前記ＡＰＩレイヤーによって、前記リクエストに関連するデータをフォーマットする工程と、

サービス層によって、前記リクエストに関連する少なくとも１つのコマンドとフォーマットされたデータを受け取る工程と、

前記サービス層によって、前記リクエストのタイプを特定する工程と、

前記リクエストの前記タイプに基づいて、前記サービス層によって、前記ブロックチェーン上で実行するための前記少なくとも１つのコマンドとフォーマットされたデータを転送する工程と、

チェーンコードによって、前記少なくとも１つのコマンドを所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように事前設定された特徴と機能に従って前記ブロックチェーンプラットフォーム上で実行可能な少なくとも１つのトランザクションに変換することによってデータ操作を管理する工程と、
前記ブロックチェーンプラットフォーム上の前記少なくとも１つのトランザクションの実行結果とフォーマットされたデータを前記クライアントアプリケーションに返す工程と
を有する、方法。
【請求項１４】
請求項１３記載の方法において、さらに、前記クライアントアプリケーションが、前記予め設定されたユースケースに従って前記ブロックチェーンプラットフォームと相互作用するために、前記事前設定された特徴と機能にアクセスするために使用できるＡＰＩを前記クライアントアプリケーションに公開する工程を有するものである、方法。
【請求項１５】
請求項１３記載の方法において、さらに、負荷分散された表現状態転送（ＲＥＳＴ）ＡＰＩを前記クライアントアプリケーションに提供する工程であって、前記ＲＥＳＴ　ＡＰＩは、前記ブロックチェーンプラットフォームと相互作用するように適合される、前記提供する工程を有するものである、方法。
【請求項１６】
請求項１３記載の方法において、さらに、
前記ＡＰＩ層、サービス層、チェーンコードを使って複数のクライアントアプリケーションが前記ブロックチェーンプラットフォーム上でデータを作成し、管理できるようにする工程を有するものである、方法。
【請求項１７】
請求項１３記載の方法において、前記リクエストに関連する前記データをフォーマットする工程は、前記ＡＰＩ層が前記クライアントアプリケーションからの前記リクエストを検証する工程と、前記クライアントアプリケーションからの前記リクエストで受け取ったユーザー詳細を検証する工程と、前記クライアントアプリケーションからの前記リクエストを実行するための前記ユーザーの許可をチェックする工程と、前記リクエストを前記ブロックチェーンプラットフォームに提出するために準備する工程とを有する、方法。
【請求項１８】
請求項１３記載の方法において、前記ＡＰＩ層、サービス層、チェーンコードは、プラットフォーム管理者、テナント、およびコアＡＳＩＫ参加者を有する３つの役割をサポートするように適合され、さらに、前記プラットフォーム管理者が、クライアントアプリケーションを登録する工程と、前記テナントが、前記テナントに割り当てられたＩＤおよびリソースを使用して前記フォーマットされたデータに対して作成、読み取り、更新、および削除の操作を実行する工程と、前記コアＡＳＩＫ参加者が、自身のＩＤを使用して前記ＡＰＩ層との要求を開始する工程とを有するものである、方法。
【請求項１９】
請求項１８記載の方法において、各テナントは、データベースを持ち、さらに、各テナントが自身のデータベースで操作を実行する工程を有するものである、方法。
【請求項２０】
請求項１３記載の方法において、さらに、
前記サービス層によって、セマンティクスとシンタックス、および少なくとも１つのコマンドとフォーマットされたデータが前記所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用する少なくとも１つのインターフェースを概説することによって、前記事前設定された特徴と機能を記述する工程を有するものである、方法。
【請求項２１】
請求項１３記載の方法において、前記所定のユースケースは、前記ブロックチェーン上で実行されるビジネスサービスに対応する、方法。
【請求項２２】

請求項１３記載の方法において、さらに、
前記チェーンコードによって、ブロックチェーン台帳を照会または更新するために前記ブロックチェーン台帳と前記ブロックチェーンプラットフォームのスマートコントラクトにアクセスする工程を有するものである、方法。
【請求項２３】
請求項２２記載の方法において、さらに、
前記クライアントアプリケーションのすべての操作を前記ブロックチェーン台帳に安全に記録する工程を有するものである、方法。
【請求項２４】
請求項２２記載の方法において、さらに、
前記サービス層が前記ＡＰＩ層からの入力を受け入れる工程と、前記チェーンコードと状態データベースを呼び出して前記入力をアクションに変換する工程とを有するものである、方法。
【請求項２５】
１若しくはそれ以上のプロセッサーによって実行される場合、前記１若しくはそれ以上のプロセッサーにブロックチェーンを管理するブロックチェーンプラットフォームにクライアントアプリケーションをインターフェースするための方法を実施させる命令を記憶した非一過性のコンピューター可読媒体であって、前記命令は、
前記クライアントアプリケーションから前記ブロックチェーン上でトランザクションを実行するためのリクエストを受け取り、前記リクエストに関連するデータをフォーマットするアプリケーションプログラミングインターフェース（ＡＰＩ）層と、
前記リクエストに関連付けられた少なくとも１つのコマンドとフォーマットされたデータを受け取り、前記リクエストのタイプを識別し、前記リクエストの前記タイプに基づいて、前記少なくとも１つのコマンドとフォーマットされたデータを前記ブロックチェーン上で実行するために転送し、前記ブロックチェーンプラットフォーム上での前記少なくとも１つのコマンドとフォーマットされたデータの実行結果を前記クライアントアプリケーションに返すサービス層と、
所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように事前設定された特徴および機能に従って、少なくとも１つのコマンドを前記ブロックチェーンプラットフォーム上で実行可能な少なくとも１つのトランザクションに変換することによってデータ操作を管理するチェーンコードと
を有する、非一過性のコンピューター可読媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本出願は、ブロックチェーンに対するモジュール式インターフェース、特に、チェーンコード上のビジネスロジックと同期したアプリケーション・プログラミング・インターフェース（ＡＰＩ）の集合体を提供することにより、ビジネスニーズに対する汎用的なソリューションを提供するアプリケーション固有統合カーネル（ＡＳＩＫ）に向けられたものである。
【背景技術】
【０００２】
ブロックチェーンは、２００９年に発明された暗号通貨と密接に関連している。しかし、ピア・ツー・ピア・マネーの誕生は、他の目的のための分散計算という全く新しいアプローチの発明でもあった。データの可用性が保証され、中央集権的な管理から解放され、暗号による改ざんが防止されるといった特徴を活用するため、さまざまな用途が模索されている。
【０００３】
ブロックチェーンの採用にまつわる現在のエネルギーは、インターネットの初期の採用を彷彿とさせる。アプリケーションやシステムが成熟するにつれ、ロジックの管理に関し

て、アプリケーションやシステムの重荷となった同じ過ちを再び繰り返すか、あるいは避ける可能性がある。初期のウェブサイトやアプリケーションと同様に、暗号通貨によるブロックチェーンの利用は、比較的単純なデータ構造と、それを管理するためのハードコードされたビジネスロジックを特徴としていた。しかし、その後「スマートコントラクト」が発明されたことで、任意の複雑さを持つルールやデータへの扉が開かれた。

【０００４】

サプライチェーンロジスティクスのような企業のユースケースは、従来のソリューションに匹敵する量の情報とロジックをブロックチェーン上に置いているが、標準化されたデータ管理ツールは何もない。すべてのプログラマーが応用コードベース内で独自のロジックセットを設計するため、その結果、ロジックは再び、それを記述するプログラム（スマートコントラクト）と緊密に結合することになる。

【０００５】

ブロックチェーンのすべての機能を企業向けアプリケーションで使用する必要があるわけではないことに留意されたい。よくある状況としては、暗号的な改ざん防止やデータの複製といった機能を実現する便利な方法としてブロックチェーンプラットフォームを利用することが挙げられる。また、企業は、将来的により分散型のアプリケーションに向けた第一歩として、コンセンサスルールを緩和したり、フロントエンドを中央集権化したりして、ブロックチェーンネットワークを自社の完全管理下で実装することもある。さらに、サポートスタッフに対する厳格なスマートコントラクトの実施とともにブロックチェーンネットワークを構築する信頼度の高いコンソーシアムは、管理者や内部監査人が自らの権限でブロックチェーンデータを直接更新できるようにするかもしれない

【０００６】

スマートコントラクトに保存された取引データと決定は、アプリケーションにおいてより大きな価値を持つ。ブロックチェーンに保存された取引データとノードの機能は、コンセンサスに支配されることなく、他の機能やロジックでその後に使用することができ、それによって操作を迅速、効率的、かつ信頼性の高いものにすることができる。必要な機能は、ブロックチェーンに保存されたデータを利用するためのアクセス権だけである。ブロックチェーンに保存された前記データは、アプリケーション、ノード機能、またはシステムによって、ブロックチェーンの内外を問わず、当初のスマートコントラクトのプログラマーが想定していた以上の目的で、効率的に読み取り、処理することができる。

【０００７】

製品開発のダイナミックな性質を考慮すると、一方的なソリューションに焦点を当てたアプリケーションは時代遅れになりつつある。これらのアプリケーションを一時停止し、アップグレードするためのコストは、多くの企業にとって大きすぎるかもしれない。頻繁なメンテナンスは、パフォーマンスの低下を招くだけでなく、経費の増加にもつながる。ロジックや製品設計図の変更は、最終的にユーザーエクスペリエンスに影響する製品のルック&フィールをしばしば変更する。ブロックチェーンのような新興技術から製品を開発するのは特に面倒で、多くのリソースを必要とする可能性がある。 分散型台帳技術のプラットフォームは、厳密なデータの複製、ルールの実施、不変性と監査可能性を提供するが、プログラミングと運用に専門的なスキルを必要とすることが多い。

【０００８】

本明細書に記載されるシステムおよび方法は、ブロックチェーンを管理するブロックチェーンプラットフォームにクライアントアプリケーションをインターフェースするアプリケーション固有統合カーネル（ＡＳＩＫ）を提供することによって、当該技術分野におけるニーズに対処する。前記ＡＳＩＫは、ブロックチェーン上でトランザクションを実行するためにクライアントアプリケーションからの要求を受け取るアプリケーションプログラミングインタフェース（ＡＰＩ）層を含む。前記ＡＰＩ層はリクエストに関連するデータをフォーマットする。前記ＡＳＩＫは、さらに、リクエストに関連する少なくとも１つのコマンドとフォーマットされたデータを受け取り、前記リクエストのタイプを識別し、リクエストのタイプに基づいて、ブロックチェーン上で実行するために前記少なくとも１つ

のコマンドとフォーマットされたデータを転送し、ブロックチェーン上の前記少なくとも１つのコマンドとフォーマットされたデータの処理結果を前記クライアントアプリケーションに返すサービス層を含む。前記サービス層は、さらに、前記アプリケーション固有統合カーネルの所定のユースケース（例えば、ビジネスサービス）に従って前記ブロックチェーンプラットフォームと相互作用するように事前設定された特徴と機能を記述する。前記ＡＳＩＫはまた、前記所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用することになる前記事前設定された特徴と機能のためのデータ操作を管理するチェーンコードも含み、このチェーンコードは、前記所定のユースケースのために前記事前設定された特徴と機能に従って前記ブロックチェーンプラットフォーム上で実行可能な少なくとも１つのトランザクションに少なくとも１つのコマンドを変換し、前記ブロックチェーンプラットフォーム上の前記少なくとも１つのトランザクションの実行結果およびフォーマットされたデータを前記サービス層に返すことによって、前記所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用することになる。

【０００９】

サンプル構成では、前記ＡＰＩ層は、前記クライアントアプリケーションが、前記アプリケーション固有統合カーネルの前記所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように前記事前設定された特徴と機能にアクセスするために使用するＡＰＩを公開する。前記ＡＰＩ層はまた、負荷分散された表現状態転送（ＲＥＳＴ）ＡＰＩを前記クライアントアプリケーションに提供してもよく、ＲＥＳＴ　ＡＰＩは、前記ブロックチェーンプラットフォームと相互作用するように適合される。前記ＡＰＩ層は、さらに、前記クライアントアプリケーションからのリクエストを検証し、前記クライアントアプリケーションからの前記リクエストで受け取ったユーザ詳細を検証し、前記クライアントアプリケーションからの前記リクエストを実行するためのユーザの許可をチェックし、前記リクエストを前記ブロックチェーンプラットフォームに提出するために準備し、前記ブロックチェーンプラットフォームから前記クライアントアプリケーションに応答を返すことができる。

【００１０】

他のサンプル構成では、前記ＡＰＩ層、サービス層、およびチェーンコードは、複数のクライアントアプリケーションが、前記ブロックチェーンプラットフォーム上でデータを作成および管理することができるマルチテナントフレームワークをサポートするように適合させることができる。前記ＡＰＩ層、サービス層、およびチェーンコードはまた、プラットフォーム管理者、テナント、およびコアＡＳＩＫ参加者を有する３つの役割をサポートするように適合させることもできる。前記プラットフォーム管理者は、クライアントのアプリケーションを担当するかもしれない。前記テナントは、フォーマットされたデータに対して作成、読み取り、更新、および削除操作を実行するために使用できるＩＤと割り当てられたリソースを持つことができる。前記コアＡＳＩＫ参加者は、前記ＡＰＩ層との要求を開始するために使用される独自のＩＤを持つことができる。また、各テナントにデータベースが提供され、各テナントがそれ自身のデータベースで操作を実行することもできる。

【００１１】

さらなるサンプル構成では、前記サービス層は、セマンティクスおよびシンタックス、ならびに少なくとも１つのコマンドおよびフォーマットされたデータが、前記アプリケーション固有統合カーネルの前記所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用する少なくとも１つのインターフェースを概説することによって、前記事前設定された特徴と機能を記述することができる。前記サービス層は、さらに、前記ＡＰＩ層からの入力を受け入れ、前記チェーンコードおよび状態データベースを呼び出すことによって入力をアクションに変換することができる。

【００１２】

さらなるサンプル構成では、前記チェーンコードは、ブロックチェーン台帳と前記ブロックチェーンプラットフォームのスマートコントラクトにアクセスして、前記ブロックチ

ェーン台帳を照会または更新することができる。前記クライアントアプリケーションのすべての操作は、前記ブロックチェーン台帳に安全に記録される。

【００１３】

本明細書は、さらに、アプリケーションプログラミングインタフェース（ＡＰＩ）層によって、前記ブロックチェーン上でトランザクションを実行するためのクライアントアプリケーションからのリクエストを受け取る工程と、前記ＡＰＩ層によって、前記リクエストに関連付けられたデータをフォーマットする工程と、サービス層によって、前記リクエストに関連付けられた少なくとも１つのコマンドおよびフォーマットされたデータを受け取る工程と、前記サービス層によって、前記リクエストのタイプを特定する工程と、前記リクエストのタイプに基づいて、前記サービス層によって、前記ブロックチェーン上で実行するために前記少なくとも１つのコマンドとフォーマットされたデータを転送する工程と、チェーンコードによって、所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように前記事前設定された特徴と機能に従って、前記少なくとも１つのコマンドを前記ブロックチェーンプラットフォーム上で実行可能な前記少なくとも１つのトランザクションに変換することによってデータ操作を管理する工程と（例えば、以下のように）、前記少なくとも１つのトランザクションの実行結果および前記ブロックチェーンプラットフォーム上でフォーマットされたデータを前記クライアントアプリケーションに返す工程と、を含む操作を実行することによって、ブロックチェーンを管理するブロックチェーンプラットフォームにクライアントアプリケーションをインターフェースするための方法に関する。

【００１４】

サンプル構成では、本方法は、さらに、前記クライアントアプリケーションが、前記所定のユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように前記事前設定された特徴と機能にアクセスするために使用することができるＡＰＩを前記クライアントアプリケーションに公開する工程を含むことができる。本方法は、さらに、負荷分散された表現状態転送（ＲＥＳＴ）ＡＰＩを前記クライアントアプリケーションに提供する工程を含んでもよく、前記ＲＥＳＴ　ＡＰＩは、前記ブロックチェーンプラットフォームと相互作用するように適合される。前記リクエストに関連するデータのフォーマッティングは、前記ＡＰＩ層が前記クライアントアプリケーションからの前記リクエストを検証する工程と、前記クライアントアプリケーションからの前記リクエストで受け取られたユーザー詳細を検証する工程と、前記クライアントアプリケーションからの前記リクエストを実行するためのユーザーの許可をチェックする工程と、前記リクエストを前記ブロックチェーンプラットフォームに提出するために準備する工程と、を含んでもよい。

【００１５】

他のサンプル構成では、本方法は、さらに、前記ＡＰＩ層、サービス層、およびチェーンコードを使用して、複数のクライアントアプリケーションが前記ブロックチェーンプラットフォーム上でデータを作成および管理できるようにする工程を含むことができる。前記ＡＰＩ層、サービス層、およびチェーンコードはまた、プラットフォーム管理者、テナント、およびコアＡＳＩＫ参加者を有する３つの役割をサポートするように適合させることができる。前記プラットフォーム管理者はクライアントアプリケーションを登録する。前記テナントは、ＩＤおよび前記テナントに割り当てられたリソースを使用して、フォーマットされたデータに対して作成、読み取り、更新、および削除の操作を実行し、前記コアＡＳＩＫ参加者は、自身のＩＤを使用して前記ＡＰＩ層へのリクエストを開始する。各テナントはまた、自身のデータベースで操作を実行することができる。

【００１６】

更なるサンプル構成では、本方法は、さらに、前記サービス層が、セマンティクス及およびシンタックス、並びに少なくとも１つのコマンドおよびフォーマットされたデータが所定のユースケースのために前記ブロックチェーンプラットフォームと相互作用する少なくとも１つのインターフェースを概説することによって、事前設定された特徴と機能を説明することを含むことができる。本方法はまた、前記サービス層が前記ＡＰＩ層からの入

力を受け入れ、前記チェーンコードおよび状態データベースを呼び出すことによって入力をアクションに変換することを含むことが可能である。

【００１７】

さらなるサンプル構成では、この方法は、前記チェーンコードがブロックチェーン台帳と前記ブロックチェーンプラットフォームのスマートコントラクトにアクセスして、前記ブロックチェーン台帳を照会または更新し、クライアントアプリケーションのすべての操作を前記ブロックチェーン台帳に安全に記録することを含むことができる。

【００１８】

本明細書には、１若しくはそれ以上のプロセッサによって実行されると、１若しくはそれ以上のプロセッサに、ブロックチェーンを管理するブロックチェーンプラットフォームにクライアントアプリケーションをインターフェース接続するための方法を実施させる命令が記憶された非一過性のコンピュータ可読媒体も含まれる。サンプル構成では、前記命令は、前記ブロックチェーン上でトランザクションを実行するために前記クライアントアプリケーションからリクエストを受信し、前記リクエストに関連するデータをフォーマットするアプリケーションプログラミングインターフェース（ＡＰＩ）層と、リクエストに関連する少なくとも１つのコマンドおよびフォーマットされたデータを受け取り、前記リクエストのタイプを識別し、前記リクエストのタイプに基づいて、前記ブロックチェーン上で実行するために少なくとも１つのコマンドおよびフォーマットされたデータを転送し、前記ブロックチェーンプラットフォーム上での少なくとも１つのコマンドおよびフォーマットされたデータの実行結果をクライアントアプリケーションに返すサービス層と、事前設定されたユースケースに従って前記ブロックチェーンプラットフォームと相互作用するように事前設定された特徴と機能に従って、少なくとも１つのコマンドを前記ブロックチェーンプラットフォーム上で実行可能な少なくとも１つのトランザクションに変換することによってデータ操作を管理するチェーンコードと、を実装する。

【００１９】

本明細書で説明する実施形態は、本開示全体を通して説明する方法を実施するためのコンピュータシステムも包含する。例えば、本明細書で説明されるシステムおよび方法は、本明細書で説明されるようなブロックチェーンプラットフォームへのデータのアクセスおよび格納のための機能を提供するために、クラウド内のコンピューティングプラットフォーム上で実装できる。

【図面の簡単な説明】

【００２０】

本開示は、添付図面の図に例示的に示されているが、これらに限定されるものではない：

【図１】図1は、例示的な例におけるＡＳＩＫ（Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｋｅｒｎｅｌ）の仕様とコアコンポーネントの概要を示す一般的なブロック図である。

【図２】図２は、サンプル構成におけるＡＳＩＫのＦａｂｌｉｃ／Ｎｏｄｅ．ｊｓ（登録商標）実装の詳細コンポーネント図である。

【図３】図３は、サンプル構成におけるＡＳＩＫ Ｔｒａｃｋアプリケーションの実装のためのＡＳＩＫ Ｔｒａｃｋシステムオブジェクトの斜視図である。

【図４】図４は、ＡＳＩＫ Ｔｒａｃｋアプリケーションのサンプル構成の制御フローを示すフロー図である。

【図５】図５は、一般的な構成におけるＡＳＩＫの動作を示すフロー図である。

【図６】図６は、ここに開示されたＡＳＩＫの１若しくはそれ以上の構成を実装するのに適した特殊用途コンピュータにプログラムされる可能性のある、典型的な汎用コンピュータのブロック図である。

【発明を実施するための形態】

【００２１】

図１－図６に関する以下の説明は、当業者がそれらを実施できるように、特定の実施形

態を十分に示している。他の実施形態は、構造的、論理的、プロセス的、および他の変更を組み込むことができる。いくつかの実施形態の部分および特徴は、他の実施形態のものに含まれてもよいし、他の実施形態のもので代替されてもよい。特許請求の範囲に記載された実施形態は、それらの特許請求の範囲の利用可能な等価物をすべて包含する。

【００２２】

ブロックチェーン：ブロックと呼ばれる継続的に増加する記録のリストは、暗号を使用してリンクされ、保護されている。各ブロックには通常、前のブロックの暗号ハッシュ、タイムスタンプ、取引データが含まれる。前記ブロックチェーンは、取引データの改ざんに対して本質的に耐性を持つように設計されている。分散型台帳として使用する場合、ブロックチェーンは通常、新しいブロックを検証するためのプロトコルを集団で遵守するピアツーピアネットワークによって管理される。一度記録されたブロックのデータは、それ以降のすべてのブロックを変更しない限り、過去にさかのぼって変更することはできない。

【００２３】

ＢＮＯ：Ｂｌｏｃｋｃｈａｉｎ Ｎｅｔｗｏｒｋ Ｏｒｃｈｅｓｔｒａｔｏｒ（ＢＮＯ）は、ＡＳＩＫ Ｔｒａｃｋ Ａｐｐｌｉｃａｔｉｏｎがブロックチェーンネットワークの設定と管理に使用するサービスである。ＢＮＯは、ユーザーがＥｔｈｅｒｅｕｍ、Ｆａｂｒｉｃネットワーク、Ｅｌｅｍｅｎｔ ＤＢインスタンスを作成するのに役立つ（Ｅｌｅｍｅｎｔ ＤＢは、組織やユーザーがブロックチェーン上のリレーショナルデータを管理する機能を提供する譲受人の製品である）。

【００２４】

チェーンコード：チェーンコードは、スマートコントラクトまたはソフトウェアで、各クライアント（組織）がブロックチェーン台帳上のデータを読み取ったり更新したりするためのビジネスロジックをサンプル構成で表したものである。ＡＳＩＫ Ｔｒａｃｋ Ａｐｐｌｉｃａｔｉｏｎでは、チェーンコードは、プラットフォーム管理者によってアップグレードされる。

【００２５】

コアＡＳＩＫ参加者：クライアントアプリケーションを介して前記ＡＳＩＫアプリケーションにアクセスしようとする人は誰でも参加者と呼ばれる。コアＡＳＩＫ参加者は独自のＩＤを持ち、それは前記ＡＳＩＫのＡＰＩ層でリクエストを開始するために前記ＡＳＩＫによって使用される。例えば、以下はここで説明される前記ＡＳＩＫ Ｔｒａｃｋの例における異なる参加者である。

サプライチェーン管理者：サプライチェーン管理者は、アプリケーションの管理者である。異なるユーザー/ユーザーグループ、ワークフローなどを作成できる。前記サプライチェーン管理者は、ワークフローや可動製品のようなすべてのエンティティを見ることができるが、それらのエンティティに対して操作を実行することはできない。

監査人：監査人は、可動製品のすべてのデータを見ることができ、紛争を解決することができる外部参加者である。

内部監査人：内部監査人は、可動製品のすべてのデータを閲覧でき、紛争を解決でき、内部参加者である。

ユーザー／参加者：ユーザーは、可動製品を作成／譲渡／受領する権限を有する。各ユーザは、複数のユーザグループに属することができ、各ユーザグループは、既存の基本製品およびワークフローに関連付けられた子製品を譲渡、受領、または拒否することができる複数のユーザを持つことができる。

【００２６】

ＣｏｕｃｈＤＢ（登録商標）は、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃにワールドステートデータベースのオプションとして組み込まれている。ＣｏｕｃｈＤＢ（登録商標）は、チェーンコードのデータ値が前記ＡＳＩＫのトランザクションデータとしてＪＳＯＮでモデル化されている場合、リッチなクエリをサポートする。

【００２７】

争議：製品の属性に食い違いがある場合、それを争議と呼ぶ。争議は、参加者全員が合意したものから逸脱するものであれば何でもあり得る。争議は、可動製品に対して、どの時点でもユーザーによって作成することができる。前記争議は、監査人またはユーザーによって解決される。

【００２８】

Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ：幅広い業界のユースケースに対応するモジュール性と汎用性を備えた、エンタープライズグレードのパーミッション付き分散型台帳プラットフォーム。

【００２９】

Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ認証局：認証局（ＣＡ）は、ブロックチェーンのユーザーに証明書サービスを提供する。具体的には、ユーザーの登録、前記ブロックチェーン上で呼び出されるトランザクション、トランスポート層セキュリティ（ＴＬＳ）で保護されたユーザー間またはブロックチェーンのコンポーネント間の接続に関するサービスである。登録認証局（ＥＣＡ）は、新規ユーザーがブロックチェーンネットワークに登録できるようにし、登録ユーザーが登録証明書ペアを要求できるようにする。１つはデータ署名用、もう１つはデータ暗号化用の証明書である。これらの証明書は、前記ブロックチェーン上でチェーンコードトランザクションを呼び出すために使用される。

【００３０】

ＪＳＯＮ：ＪａｖａＳｃｒｉｐｔ Ｏｂｊｅｃｔ Ｎｏｔａｔｉｏｎの略で、軽量でテキストベースの、人間が読めるデータ交換フォーマット。

【００３１】

可動製品：可動製品は、あるノードから別のノードに転送される実際の製品である。ユーザーは、可動製品を作成、転送、受領、更新、および拒否できる。

【００３２】

Ｎｅｓｔ（ＮｅｓｔＪＳ）：効率的でスケーラブルなＮｏｄｅ．ｊｓ（登録商標）サーバーサイドアプリケーションを構築するためのフレームワークである。ＮｅｓｔＪＳは、プログレッシブＪａｖａＳｃｒｉｐｔを使用し、ＴｙｐｅＳｃｒｉｐｔでビルドされ、完全にサポートされている（それでも、開発者は純粋なＪａｖａＳｃｒｉｐｔでコーディングすることができる）。Ｎｅｓｔ．ＪＳの詳細については、「ＮｅｓｔＪＳについて、ドキュメントと基盤」を参照されたい。（ｈｔｔｐｓ：／／ｎｅｓｔｊｓ．ｃｏｍ／）

【００３３】

Ｎｏｄｅ．ｊｓ（登録商標）：ブラウザの外でＪａｖａＳｃｒｉｐｔコードを実行する無償のオープンソースサーバ環境。Ｎｏｄｅ．ｊｓ（登録商標）は、ＣｈｒｏｍｅブラウザのＪａｖａ Ｓｃｒｉｐｔランタイム上に構築された非同期プログラミングプラットフォームである。Ｎｏｄｅ．ｊｓ（登録商標）の詳細については、「Ｎｏｄｅ．ｊｓ（登録商標）について」を参照されたい。（Ｎｏｄｅ．ｊｓ，Ｎｏｄｅ．ｊｓ Ｆｏｕｎｄａｔｉｏｎ）

【００３４】

プラットフォーム管理者：プラットフォーム管理者は、製品展開時に登録される。このために特別に設計されたプラットフォーム管理者の役割は、ＡＳＩＫ内のテナントを作成および管理する権限のみを持っている。

【００３５】

製品：製品は、すべての可動製品のメタデータである。１つの製品に対して複数の可動製品を作成することができる。

【００３６】

スマートコントラクト：前記ブロックチェーンまたは分散型台帳上で行われる汎用計算を提供するコンピュータプロトコル。スマートコントラクトトランザクションは、シンプルな場合もあれば、高度なロジックを実装する場合もある。結果として生じる取引は通常、台帳参加者には透過的で、追跡可能で、不可逆的である。

【００３７】

テナント：テナントは、前記ＡＳＩＫのプラットフォーム管理者によって登録される。テナントは、前記ＡＳＩＫを利用するあらゆるクライアント（組織）を表す。前記ＡＳＩＫは、複数のクライアントに１つのアプリケーションを利用させる機能を提供し、これをマルチテナントと呼ぶ。テナントは、ＩＤとそれに割り当てられたリソースを持ち、前記ＡＳＩＫ内の任意のアセットに対して必要なＣＲＵＤ（作成、読み込み、更新、削除）操作を実行するために使用することができる。

【００３８】

トラッカー：各可動製品について、１つのＴｒａｃｋｅｒドキュメントが作成される。このドキュメントには、可動製品の属性更新や移動を含む、前記可動製品のすべての履歴情報が含まれる。

【００３９】

ワークフロー：ワークフローは、製品が転送されるユーザーグループの順序と、ベース製品に関連する属性を決定する。

【００４０】

コンピュータアプリケーションの設計者、特に大規模な企業環境では、既存の分散型台帳技術（ＤＬＴ）プラットフォームをソリューションに組み込むケースが増えている。上述のように、これらのプラットフォームは厳密なデータ複製、ルール実施、不変性と監査可能性を提供するが、多くの場合、プログラミングと運用に専門的なスキルを必要とする。本明細書で説明するように、近接するＤＬＴノードの通信の抽象化（そのデータに対するデータとルールの両方の書き込みを含む）と、カテゴリ内（例えば、特定のユースケース）の潜在的なＤＬＴアプリケーションの広範な範囲をサポートするデータおよびルールプリミティブのセットを組み合わせることにより、選択したプラットフォームの専門スキルを持たない開発者によるクライアントアプリケーションの作成と運用を可能にするサブシステムを使用することができる。プリミティブは通常、アプリケーション開発者が特定のユースケースに適合させ、ＤＬＴプラットフォームのスキルを必要とせず、ＤＬＴプラットフォームが提供する厳格なルール適用を失うことなく、属性や追加データルールで拡張できるように構築されている。また、クライアントアプリケーション開発者が、これらのプリミティブを設計し利用するためのインターフェースは、前記ＤＬＴプラットフォームの技術的理解の必要性を完全に排除している。

【００４１】

具体的な実装は、単一のクライアントアプリケーションの専用サブシステム、多数のクライアントアプリケーションで使用する再利用可能なコンポーネント、または多数のクライアントアプリケーションにアプリケーションイネーブルメントを提供するサービスである。他の実装は、１若しくはそれ以上の近接したＤＬＴノードとの通信を仲介し、これらのノードは単一または複数のＤＬＴネットワークから引き出される。

【００４２】

本明細書では、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃに関して説明するが、本明細書で説明する技術は、ＡＩＯＮ、ＡｒｃＢｌｏｃｋ、ＥＯＳ、ＮＥＯ、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｓａｗｔｏｏｔｈ、Ｎｘｔ、ＱＴＵＭ、Ｑｕｏｒｕｍ、Ｓｍｉｌｏ、Ｔｅｚｏｓ、ＴＲＯＮ、Ｗａｃｈａｉｎ、またはＺｉｌｌｉｑａなどの他のブロックチェーンネットワークでも使用できることが当業者には理解されるであろう。本明細書で使用される用語は、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃブロックチェーンネットワークに関するものである。対応する用語は、言及したような他のブロックチェーンネットワークの文脈で使用される。

【００４３】

サンプルサブシステムは、広範なカテゴリーまたは仮想的なワークフローと情報共有アプリケーション、およびそのようなクライアントアプリケーションのいくつかの運用上の実世界での実現に対して、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ　ＤＬＴプラットフォームの間接的な使用を可能にすることができる。このようなサブシステムを実装するには、３つの重要な要件がある：

１）前記分散台帳上で表現される、拡張可能な汎用データとアクションプリミティブのセットと、
２）前記分散台帳に埋め込まれた、これらのプリミティブ上で動作するルールセットであって、前記サブシステムの固有データルールと、前記クライアントアプリケーションによって提供される可能性のある、これらのルールの許容される拡張の両方を強制する、ルールセットと、
３）前記ブロックチェーンプラットフォームの知識がなくても操作できるプログラミングインターフェース、である。
その結果、サブシステムは、あるアプリケーションカテゴリーの知識および／またはあるブロックチェーンプラットフォームのスキルを持つ一般ユーザーが、そのカテゴリーのクライアントアプリケーションの作成をサポートする独自のサブシステムを構築することを可能にする一連のツールとライブラリを提供する。結果として得られるコードは、ＡＰＩネットワークとブロックチェーンプロトコルとの間のミドルウェアとして実装され、前記ユーザーは前記ブロックチェーンの操作に関する広範な知識がなくても、一貫して前記ブロックチェーンにインターフェースすることができる。このようなアプリケーションでは、上位層のユースケース固有のビジネスルールが記述され、前記ブロックチェーン上で強制可能である。前記ブロックチェーンの機能と制御を中断することなく、ワークフローを更新するために新しいビジネスルールを随時追加することができる。

【００４４】

本書で説明するＡＳＩＫ（Ａｐｐｌｉｃａｔｉｏｎ Ｓｐｅｃｉｆｉｃ Ｉｎｔｅｇｒａｔｅｄ Ｋｅｒｎｅｌ：アプリケーション固有統合カーネル）は、主要なビジネスニーズに対する汎用的なソリューションを提供するモジュール式インターフェースである。本明細書で説明するように、ＡＳＩＫはチェーンコード上のビジネスロジックと同期したＡＰＩの集合体であり、「プラグアンドプレイ」の性質を与え、物事を非常にシンプルに保つ。前記ＡＳＩＫのウェブサービス層は、ビジネス要件を特定し、タスクや操作を実行するために必要なバンドルから、表現状態転送（ＲＥＳＴ）ＡＰＩを公開する。この機能により、前記ＡＳＩＫはほとんどどのような業種でも実装できる汎用的なソリューションとなる。以下の説明から明らかなように、前記ＡＳＩＫは、資産追跡、サプライチェーン管理、同意管理、金融ソリューション、アクセス制御、製品在庫、その他多くのアプリケーションで実装される可能性のあるマルチユーティリティアプリケーションである。

【００４５】

一般的に、前記ＡＳＩＫは、前記ブロックチェーン上で頻繁に実行される本質的なユースケースのためのＡＰＩの集合体である。エンドユーザーによって行われる特定のトランザクションリクエストは、前記リクエストから収集された入力データによって構成される、あらかじめ想定されたシナリオの一般的な集合体によって実行される。前記トランザクションが、前記ブロックチェーン上で成功裏に実行されると、レスポンスは実行可能なアウトプットの形で前記ユーザーに戻る。

【００４６】

前記ＡＳＩＫの価値は、前記ブロックチェーンを理解しなくてもすぐに使えることにある。さらに、前記ＡＳＩＫはあらゆる主要なビジネス要件を満たすことができる汎用的なソリューションを提供する。前記ＡＳＩＫは高度に「プラグイン可能で再利用可能」であるため、ユーザーは前記ＡＳＩＫを要件に合わせて使用するために設定する要素を最小限に抑えることができる。前記ＡＳＩＫは汎用的なソリューションを提供するため、市場の成長や急速に変化する要件に合わせて進化する。前記ＡＳＩＫは、複数の外部アプリケーションと統合して使用することができ、マルチユーティリティアプリケーションとなっている。また、前記ＡＳＩＫは常に「進化」しており、常に新機能やアップグレードの余地があるため、高速で超近代的なソリューションとなっている。前記ＡＳＩＫは、ユーザーと前記ブロックチェーンの間の信頼できるインターフェイスであるという点で、「前記ブロックチェーンに対するブラックボックス」でもある。前記ＡＳＩＫが提供するメリットを享受するために、ユーザーはブロックチェーンの概念を理解する必要はない。前記ＡＳ

ＩＫはアクセスが容易なため、外部システムと難なく統合することができる。前記ＡＳＩＫは、開発をサポートするほとんどすべての認識可能なブロックチェーンネットワークで機能する。また、前記ＡＳＩＫは、クラウドベースのプラットフォームと互換性があるため、汎用性が高く、導入や統合が容易である。

【００４７】

ＡＳＩＫは、以下のようなさまざまなアプリケーションに実装することができる：

【００４８】

ＡＳＩＫ　Ｔｒａｃｋ：全参加組織のためのマーケットプレイスを内蔵した、洗練された自律的なフィンテック対応情報サプライチェーンを構成・展開するための組織を支援するプラットフォーム。ＡＳＩＫ　Ｔｒａｃｋの実施例を以下に説明する。

【００４９】

ＡＳＩＫエコシステム：ピアツーピアの銀行取引を可能にするプラットフォーム。分散型台帳技術を使用して構築されているため、前記プラットフォーム上のすべての取引は１００％監査可能で、不変であり、許可されたユーザーのみが見ることができる。

【００５０】

ＡＳＩＫ　Ｃｏｎｓｅｎｔ：データ収集の全く新しい方法を定義し、データが使用されるたびにユーザーに通知する、完全に実行可能なデータ同意管理プラットフォーム。

【００５１】

ＡＳＩＫのサンプル実施形態

前述したように、一方的なソリューションに焦点を当てただけのアプリケーションは時代遅れになりつつあり、一時停止してアップグレードするプロセスは、単にパフォーマンスの低下と費用の増加を招くだけである。ブロックチェーンのような新たな技術に根ざしつつ、普遍的で非常に使いやすいビジネスソリューションが求められている。しかし、影響力のある技術を使ってそのような製品を開発するのは面倒で、多くのリソースを必要とする。別の技術スタックに移行することはかなりの難題であり、多くの企業にとって最も実行可能な選択肢ではない。

【００５２】

本明細書で説明するＡＳＩＫは、ブロックチェーンだけでなく非ブロックチェーンアプリケーションでも活用できる再利用可能でプラグイン可能なサービスを提供することによって、当該技術分野におけるこれらのニーズに対処する。本明細書で説明する前記ＡＳＩＫは、様々なブロックチェーンプラットフォームとユーザプログラミング言語にわたって実装できるように設計されている。システムの一部はカスタマイズされてもよいが、一連の共通仕様に従っている。この説明では、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）ＦａｂｒｉｃプラットフォームとＮｏｄｅ．ｊｓ（登録商標）プログラミング言語に基づく、そのような実装の１つの説明を提供する。しかし、本明細書で提供される説明は、当業者によって他のプラットフォームおよび言語に容易に拡張され得ることが理解されよう。

【００５３】

ＡＳＩＫの全実装のための一連の仕様、インターフェース、およびコアコンポーネントの説明は、図１に関して提供される。

【００５４】

図１は、例示的な実施例におけるＡＳＩＫ（Ａｐｐｌｉｃａｔｉｏｎ　Ｓｐｅｃｉｆｉｃ　Ｉｎｔｅｇｒａｔｅｄ　Ｋｅｒｎｅｌ：アプリケーション固有統合カーネル）１００の仕様と構成要素の概要を示す一般的なブロック図である。図１は、サンプル構成におけるＡＳＩＫの構成要素と仕様を示す適合構成要素図である。前記共通仕様は、抽象的な要件を示す「仕様」ステレオタイプを持つ構成要素として表される。仕様は、すべてのＡＳＩＫ実装の間で共通の方法で実装されることが期待される側面である。示されたインターフェースも、すべての実装に共通の仕様である。

【００５５】

図１において、クライアントアプリケーション１１０$_1$－１１０$_N$は、前記ＡＳＩＫ１００を利用しようとするアプリケーションである。前記クライアントアプリケーション１

１０$_1$－１１０$_N$は、任意のサードパーティアプリケーション、サービス、またはツールであってもよい。前記ＡＳＩＫ１００にアクセスする複数のクライアントアプリケーション１１０$_1$－１１０$_N$またはテナントが存在する可能性がある。前記クライアントアプリケーション１１０$_1$－１１０$_N$は、ＡＳＩＫテナントオンボードサービス１２０を介して前記ＡＳＩＫ１００にアクセスし、複数のクライアントまたはテナントを処理し、すべてのテナントに前記ＡＳＩＫ１００を効率的に使用させる。前記ＡＳＩＫテナントオンボードサービス１２０は、ＡＳＩＫウェブサービスインターフェース１２５を介してＡＳＩＫウェブサービス１３０と相互作用する。ＡＳＩＫウェブサービス１３０は、ＨＴＴＰリクエストからコマンドを受け取り、前記ＡＳＩＫ１００の内部構成要素と通信して所望の結果を生成する構成要素である。前記ＡＳＩＫウェブサービス１３０は、前記クライアントアプリケーション１１０$_1$－１１０$_N$と直接接続するサービスを提供し、要求に基づいて、リクエストをサービスレイヤの他のサービスに転送することができる。したがって、前記ＡＳＩＫウェブサービス１３０は、データの初期処理を行い、リクエストのタイプを識別し、前記タイプに基づいて、前記リクエストをさらに転送して処理を完了し、その結果（中間または最終）を前記クライアントアプリケーション１１０$_1$－１１０$_N$に返すマスターサービスと考えることができる。ＡＳＩＫ　ＡＰＩ層１４０は前記ＲＥＳＴコールを処理し、データを操作／フォーマットして他の様々な構成要素にデータを送信するために前記ＡＳＩＫウェブサービス１３０にも実装される。

【００５６】

前記ＡＳＩＫウェブサービス１３０は、前記ＡＳＩＫサービス層インタフェース１４５を介して前記ブロックチェーンプラットフォーム固有のＡＳＩＫサービス層１５０と相互作用する。前記ブロックチェーンプラットフォームＡＳＩＫサービス層１５０は、ブロックチェーンシステム（この例ではＨｙｐｅｒｌｅｄｇｅｒ　Ｆａｂｒｉｃ）で動作するように設計されており、前記ＡＳＩＫウェブサービス層１３０を介して前記ＡＳＩＫ　ＡＰＩ層１４０からの入力を受け入れ、ビジネスロジックを実行する。前記ＡＳＩＫサービス層１５０は、ＤＬＴライブラリー１５５や外部サービス１６０を介して利用可能なものなど、１若しくはそれ以上の相互接続されたサービスを含むことができる。前記ＤＬＴライブラリー１５５と外部サービス１６０は、例えばＡｐａｃｈｅ　Ｋａｆｋａキューを使用して相互に通信する。前記ブロックチェーンプラットフォームＡＳＩＫサービス層１５０は、インターフェースと、前記ＡＳＩＫ１００のあらゆる実現（ユースケース）でサポートされる機能を記述する。前記ブロックチェーンプラットフォームＡＳＩＫサービス層１５０の仕様は、インターフェース全体をカバーする以外に、前記ブロックチェーンプラットフォームＡＳＩＫサービス層１５０または前記ブロックチェーンプラットフォーム１８０自体のいずれかで実現され得る特徴と機能のサブセットもカバーする。

【００５７】

前記ブロックチェーンプラットフォームＡＳＩＫサービス層１５０は、ＡＳＩＫウェブサービスＡＰＩ層１４０からの入力を受け入れ、以下の工程を実行する：

１）受け取ったリクエストを検証し、秘密にするべき情報を除去する。

２）リクエストで受け取ったユーザーとテナントの詳細を検証する。

３）リクエストされた操作を実行するためのユーザーの権限をチェックする。

４）前記ブロックチェーンプラットフォーム１８０に提出するトランザクション／リクエストオブジェクトを準備する。

５）前記ブロックチェーンプラットフォーム１８０にトランザクションを提出する。

６）工程５から返されたレスポンスに基づいて、成功／エラーレスポンスオブジェクトを準備する。

７）クライアントにレスポンスを返す。

前記ブロックチェーンプラットフォームＡＳＩＫサービス層１５０は、サンプル構成ではＮｏｄｅ．ｊｓ（登録商標）で実装されており、プログラミング言語に依存しないため、対象とするブロックチェーンプラットフォーム１８０の任意の言語で実装することができる。前記データが所望の形式に変換されると、前記ブロックチェーンプラットフォーム

ＡＳＩＫサービス層１５０はインターフェース１６５を介してブロックチェーンプラットフォーム固有ＡＳＩＫチェーンコード１７０を呼び出す。前記ブブロックチェーンプラットフォーム固有ＡＳＩＫチェーンコード１７０は、今度は前記ブロックチェーンプラットフォーム１８０を呼び出す。追加の機能は、ブロックチェーンプラットフォーム１８０自体をターゲットにしてもよい。

【００５８】

サンプル構成では、前記ＡＳＩＫ１００のデータ操作は、前記ＡＳＩＫチェーンコード１７０と前記ブロックチェーンプラットフォーム１８０が一緒に管理する。正確な作業分担は前記ブロックチェーンプラットフォーム１８０の能力に依存する。必要な機能は２つのサブ仕様によって与えられる。ブロックチェーン上のトランザクションデータ１８５は、サポートされる標準化されたデータ表現（ＪＳＯＮスキーマ）とデータ操作を定義する。前記トランザクションデータには、ワークフロー、移動可能な製品、製品、サプライチェーン管理者、監査人、内部監査人、トラッカー、ユーザー、ユーザーグループ、争議などのタイプの文書が含まれる。台帳更新１９０は、前記ブロックチェーン上でトランザクションが実行されるたびに、企業によって行われたすべての前記トランザクションを追跡する前記台帳上で更新される一意のＩＤを持つトランザクションを提供する。

【００５９】

ＡＳＩＫ　１００の運用中、前記ＡＳＩＫテナントオンボードサービス１２０は、単一の物理サーバー上で実行されるソフトウェアの複数のインスタンスをサポートし、複数のユーザーが単一のアプリケーション、例えばデータベースや特定のマイクロサービスを使用することをサポートするマルチテナントフレームワークを提供する。図１に見られるように、複数のクライアントアプリケーション（テナント）$110_1$－$110_N$は、前記ＡＳＩＫ　１００にアクセスしようとするかもしれない。前記ＡＳＩＫテナントオンボードサービス１２０は、これらの複数のテナントがすべて単一の前記ＡＳＩＫ　１００を使用できるように処理する。前記マルチテナント機能を組み込むことにより、前記ＡＳＩＫ　１００は、リソースの共有がシステムコストを削減するという点で費用対効果が高い。また、すべてのユーザーが同じ技術プラットフォームからサービスにアクセスするため、自動更新や頻繁な更新にアクセスするのがより簡単になる。前記ユーザーは個別のリソース割り当ての料金を支払う必要がない。ホスティングも簡素化され、ハードウェアに依存しない。マルチテナントは、同じインフラやデータセンターに常駐する能力を提供し、サーバーやコンピューティング容量を容易に追加することができる。マルチテナントはまた、悪意のあるソフトウェアにさらされる機会を減らし、手間のかからないアップグレードも提供する。また、前記マルチテナント機能により、仮想化とリモートアクセス機能を企業内で完全に使用することができる。

【００６０】

前記マルチテナントの作成は、前記ブロックチェーンプラットフォーム１８０（Ｆａｂｒｉｃなど）のリソースの割り当てによって指示され、この割り当ては専用または共有にすることができる（クライアント／ユーザーの要求に依存する）。マルチテナント作成はまた、チャネル作成とスマートコントラクトの展開、ＣｏｕｔｈＤＢ（登録商標）ビュー作成、Ｋａｆｋａトピックパーティション作成、およびＥｌａｓｔｉｃｓｅａｒｃｈインデックス作成によって指示されることもある。

【００６１】

図１に関して上述したように、ＡＳＩＫ　１００の主要構成要素の１つは、前記ＡＳＩＫ　１００のすべての実現で使用される前記ＡＳＩＫウェブサービス１３０である。前記ＡＳＩＫウェブサービス１３０は、前記クライアントアプリケーション$110_1$－$110_N$と前記ＡＳＩＫ　１００の間のインターフェースとして機能する。ユーザーは、ソフトウェア開発キット（ＳＤＫ）またはアプリケーションへのインタフェースを介して、あるいはＨＴＴＰＳリクエスト（またはＲＥＳＴｆｕｌ　ＡＰＩからのリクエスト）を介してリクエストを行うことができる。これらのリクエストはすべて、前記ＡＳＩＫウェブサービス１３０のサブ構成要素を介して（他の構成要素に）受信、処理、転送される可能性が

ある。そのようなサブ構成要素の１つが前記ＡＳＩＫ　ＡＰＩ層１４０であり、許容可能な構文の形式で入力を受け付ける。ミドルウェア（図４－５）は、すべてのパラメータがビジネス要件に沿っていることを確認するために、これらの構文を解析、検証、およびフォーマットする重い作業を行い、次に構文を前記ＡＳＩＫサービス層１５０および他の後続構成要素にルーティングする。

【００６２】

ビジネスニーズに応じて、要求に応える適切な機能が前記ＡＳＩＫ　１００によって選択され、前記サービス層がそれらの機能を提供する。前記ブロックチェーンプラットフォーム固有の前記ＡＳＩＫサービス層１５０はプログラミング言語に依存しないため、どの言語でも実装可能である。前記サービスは、Ｎｏｄｅ．ｊｓ（登録商標）とＮｅｓｔＪＳを用いて開発される。前記ブロックチェーンプラットフォーム固有前記ＡＳＩＫサービス層１５０は再利用可能であり、前記ブロックチェーンフレームワークに依存しない。前記ブロックチェーンプラットフォーム固有ＡＳＩＫサービス層１５０の仕様は、ＡＳＩＫ　１００のすべての実現においてサポートされる必要のあるインターフェースと機能を記述している。前記ブロックチェーンプラットフォーム固有前記ＡＳＩＫサービス層１５０は、ｈｔｔｐリクエスト、ｋａｆｋａ、またはその他のプロトコルを介して相互接続された複数のサブサービスを含むことができる。さらに、複数のＤＬＴライブラリ１５５（例えば、ｄｌｔ－ｋａｆｋａ）も、既存の機能の高い再利用性を達成するために、前記ＡＳＩＫ　１００の前記ブロックチェーンプラットフォーム固有前記ＡＳＩＫサービス層１５０に実装することができる。例えば、ｄｌｔ－ｋａｆｋａは、ｋａｆｋａに接続するメカニズムを提供することで、ｐｒｏｄｕｃｅ／ｃｏｎｓｕｍｅのような機能を実装するのではなく、どのサービスでも使用できるように公開する。ｒｅｄｉｓ、ｅｌａｓｔｉｃｓｅａｒｃｈ、ＣｏｕｃｈＤＢ（登録商標）などの外部サービス１６０への接続はすべて、前記ブロックチェーンプラットフォーム固有前記ＡＳＩＫサービス層１５０自体に確立することができる。

【００６３】

前記ブロックチェーンプラットフォーム固有前記ＡＳＩＫサービス層１５０の前記リクエスト機能は、次に前記ＡＳＩＫチェーンコード１７０の同様の機能をトリガーする。この関数は、入力リクエストを前記ＡＳＩＫチェーンコード１７０を介して前記ブロックチェーンプラットフォーム１８０上で実行可能なトランザクションに変換する変調器として機能する。このビジネス上の問題は、前記ブロックチェーンプラットフォーム１８０とのすべてのやり取りを担う前記ＡＳＩＫチェーンコード１７０において解決される。

【００６４】

プラットフォームそのものと同様に、前記ブロックチェーンプラットフォーム１８０に直接実装される機能の実装も、アプローチが大きく異なる。対応する仕様は、使用される前記ブロックチェーンプラットフォーム１８０に関係なく、依然として共通となるものを定めている。前記ブロックチェーンプラットフォーム１８０上でトランザクションが実行されるたびに、前記トランザクションのステータス、すなわち成功か失敗かにかかわらず、企業によって行われたすべてのトランザクションを追跡する台帳１９０上で一意のＩＤを持つトランザクションが更新される。前記スマートコントラクトおよび／またはチェーンコード１７０は、前記ＡＳＩＫ　１００によってカスタマイズされ、トランザクションレポートをさらに強化することができる。ビジネス操作が完了すると（成功シナリオ）、前記トランザクションの状態は、データの現在の状態を保持する事前に構成された状態データベース上に更新される。前記ＡＳＩＫ　１００は状態データベースとしてＣｏｕｃｈＤＢ（登録商標）を使用することができ、ＪＳＯＮ形式でのデータの保存、データに対するＪＳＯＮクエリの発行、およびクエリをサポートするためのインデックスの使用を可能にする。一方、トランザクションが失敗するたびに、エラーハンドリングシステムは前記トランザクションを適切にＡＳＩＫ　１００に割り当てることができる。

【００６５】

図２に示すサンプル構成では、Ｎｏｄｅ．ｊｓ（登録商標）、Ｎｏｄｅ．ｊｓ（登録商

標）ＳＤＫに基づくＨｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃブロックチェーンプラットフォーム、およびＡｐａｃｈ　ＣｏｕｃｈＤＢ（登録商標）を使用して、前記ＡＳＩＫ　１００の構成が実装されている。Ｎｏｄｅ．ｊｓ（登録商標）は現在、ウェブサーバーサイド開発のための世界で最も人気のあるフレームワークであり、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃは前記ＡＳＩＫ　１００の多くの要件を部分的に満たすアーキテクチャを持っています。Ａｐａｃｈ　ＣｏｕｃｈＤＢ（登録商標）は、世界状態データベースのオプションとしてＨｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃに組み込まれています。ＣｏｕｃｈＤＢ（登録商標）は、チェーンコードのデータ値がＡＳＩＫデータとしてＪＳＯＮでモデル化されている場合、リッチなクエリをサポートする。

【００６６】

上述したように、前記ＡＳＩＫ　１００は事前に設定されたシナリオを備えた「プラグアンドプレイ」ソリューションである。リクエストは、従来のユーザーインターフェース（ＵＩ）メソッドまたはＨＴＴＰリクエストを使用して、クライアントアプリケーション$110_1$－$110_N$から前記ＡＳＩＫウェブサービス１３０に供給される。取引データ機能を利用可能にするため、スマートコントラクトも前記ブロックチェーンプラットフォーム１８０に実装されている。Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃはブロックチェーン技術の実装であり、Ｌｉｎｕｘ　Ｆｏｕｎｄａｔｉｏｎの下で共同開発されている。前記Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃブロックチェーンプラットフォームは、上述のように仕様の要件をすべて満たし、さらに優れたパフォーマンス、テナントとアプリケーション間の論理的なデータ分離、前記ブロックチェーンプラットフォーム１８０との緊密な統合を提供するため、適している。

【００６７】

図２は、サンプル構成における前記ＡＳＩＫ　１００のＦａｂｒｉｃ／Ｎｏｄｅ．ｊｓ（登録商標）実装２００の詳細コンポーネント図である。図２は、Ｎｏｄｅ．ｊｓ（登録商標）がプログラミング言語であり、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃが使用されるブロックチェーンプラットフォームであるＡＳＩＫ　Ｔｒａｃｋ構成２００の実装を説明する。図２は、Ｆａｂｒｉｃ／Ｎｏｄｅ．ｊｓ（登録商標）実装が、いくつかの追加機能とともに、図１で概説した仕様をどのように実現するかを示している。図２から、Ｆａｂｒｉｃ上のＡＳＩＫデータ実現には、ＡＳＩＫ実装２００にプラットフォーム管理者、テナント、およびコアＡＳＩＫ参加者の合計３つの役割を提供することが含まれることが理解されよう。プラットフォーム管理者は、製品展開時に登録される。この役割は、テナントを前記システムに登録する責任を負う。テナントは、ＩＤとそれに割り当てられたリソースを持ち、それを使って前記ＡＳＩＫ実装２００のあらゆるアセットで必要なＣＲＵＤ操作を行うことができる。コアＡＳＩＫ参加者は独自のＩＤを持ち、これは前記ＡＳＩＫ　１００がＡＳＩＫのＡＰＩ層１４０との要求を開始するために使用される。例えば、ＡＳＩＫ　Ｔｒａｃｋ構成では、少なくとも４つの異なる参加者が存在する可能性がある：サプライチェーン管理者、監査人、内部監査人、ユーザー／参加者などである。

【００６８】

ＡＳＩＫ　Ｔｒａｃｋアプリケーション１１０は、ＡＳＩＫオンボードサービスＡＰＩインターフェース２１０を介してアクセスされるＡＳＩＫ　Ｔｒａｃｋテナントオンボードサービスコードとデータ１２０´を含む前記ＡＳＩＫ　Ｔｒａｃｋテナントオンボードサービス１２０の助けを借りて、マルチテナントシステムを考慮する。ＡＳＩＫオンボードサービス２１０は、複数のクライアント（テナント）が、前記テナントにサービス操作を知られることなく単一のアプリケーションを使用することを可能にする。内部的には、前記ＡＳＩＫ　Ｔｒａｃｋテナントオンボードサービス１２０は、テナントごとに異なるデータベースを作成し、テナントの詳細を区別できるようにする。各テナントは、それを意識することなく、独自のデータベースで独自の操作を実行する。マルチテナントを使用することにより、１つのアプリケーションを作成し、その後、エンドユーザーごとに個別のソリューションを再作成する必要がないように、希望する多くの顧客に展開することが

できる。このように、テナントにとって、前記アプリケーションは単一のユニットであり、すべての内部パーティションはブラックボックスである。

【００６９】

前記ＡＳＩＫウェブサービスＡＰＩ層１３０は、クライアントが直接やりとりするサービスであるため、製品の重要な部分である。ＲＥＳＴ　ＡＰＩの助けを借りて、前記ＡＳＩＫウェブサービスＡＰＩ層１３０はリクエストを受け取り、ｅｘｐｒｅｓｓ－ｖａｌｉｄａｔｏｒのようなＪａｖａＳｃｒｉｏｐｔ実行環境Ｎｏｄｅ．ｊｓ（登録商標）用のノードパッケージ管理（ＮＰＭ）モジュールを使用して妥当性確認／秘密にするべき情報の除去を実行する。前記ＮＰＭモジュールは、Ｎｏｄｅ．ｊｓ（登録商標）アプリケーションによってインポートされ、前記ＡＳＩＫウェブサービスＡＰＩ層１３０とのインターフェース接続を確立し、前記インターフェース接続されたクライアントアプリケーション１１０からのルール実行要求を開始する。

【００７０】

データがフォーマットされると、前記データは、前記ＡＳＩＫウェブサービスＡＰＩ層１４０である次の層に転送される。前記ＡＳＩＫウェブサービスＡＰＩ層１４０は、本明細書では前記ＡＳＩＫ　Ｔｒａｃｋのウェブサービス内のサービス、および外部のＡＳＩＫ　Ｔｒａｃｋサービス（ウェブサービス外のサービス）と呼ばれる。リクエストのタイプを識別し、適切なサービスに転送する仕事は、前記ＡＳＩＫウェブサービスＡＰＩ層１４０が行う。前記リクエストの一部は、前記ＡＳＩＫウェブサービスＡＰＩ層１４０によって完全に処理されるが、他の処理については、前記ＡＳＩＫウェブサービスＡＰＩ層１４０の下にあるバッチコンポーザーサービスやアグリゲーションエンジンサービスなどの前記ＡＳＩＫ　Ｔｒａｃｋの他のサービスの助けを借りる。このようなサービスはウェブサービスではないが、要求に応じてウェブサービスを通じて呼び出される。

【００７１】

例えば、ワークフロー（ＡＳＩＫ　Ｔｒａｃｋのエンティティ）を作成するために必要なのは前記ウェブサービスだけなので、呼び出しは容易である。しかし、アグリゲーション（ＡＳＩＫ　Ｔｒａｃｋの操作）を実行するために、前記ＡＳＩＫ　Ｔｒａｃｋウェブサービスは必要なリクエスト検証を行い、検証に成功すると、アグリゲーション機能（ｃｏｕｎｔ、ｍａｘ、ｍｉｎ、ａｖｅｒａｇｅ、ｓｕｍ）を実行するためにＴｒａｃｋのアグリゲーションエンジンサービスに前記リクエストを転送する。前記ＡＳＩＫウェブサービスＡＰＩ層１３０のサービスは、ＲＥＳＴ　ＡＰＩまたはＡｐａｃｈｅ　Ｋａｆｋａを使用して相互に接続する。各サービスは、ジョブの一部を指定され、前記サービスは、それを完了し、残りの前記ジョブを完了するためにさらに前記リクエストを送信する。

【００７２】

前記ＡＳＩＫテナントオンボードサービス１２０、ＡＳＩＫウェブサービスＡＰＩ層１３０、ＡＳＩＫウェブサービスＡＰＩ層１４０、ブロックチェーンプラットフォーム固有ＡＳＩＫサービス層１５０、ＡＳＩＫチェーンコード１７０、およびブロックチェーンプラットフォーム１８０、ならびに前記インターフェース１２５、１４５、および１６５は、すべての実装に共通であり、図１と同一である。その下にある前記ブロックチェーンプラットフォーム固有Ｎｏｄｅ．ｊｓ（登録商標）Ｆａｂｒｉｃサービス層１５０´は、すべてのプラットフォーム操作のためのファサードを実現する。ブロックチェーンプラットフォーム仕様１８０への依存を通じて前記ブロックチェーンプラットフォーム固有Ｎｏｄｅ．ｊｓ（登録商標）Ｆａｂｒｉｃサービス層１５０´を直接サポートするのは、特定のインデックス作成機能のために直接アクセスされる前記Ｆａｂｒｉｃ　ＣｏｕｃｈＤＢ実装２２０である。Ｆａｂｒｉｃの実現では、スキーマインデックスの変更とユーザー権限は、部分的に前記ブロックチェーンプラットフォーム固有Ｎｏｄｅ．ｊｓ（登録商標）Ｆａｂｒｉｃサービス層１５０´に実装され、前記Ｆａｂｒｉｃ　ＣｏｕｃｈＤＢ（登録商標）実装２２０を直接呼び出す。

【００７３】

前記Ｆａｂｒｉｃ　ＣｏｕｃｈＤＢ（登録商標）機能２２０は、前記ブロックチェーン

プラットフォーム１８０への依存を通じて、前記ブロックチェーンプラットフォーム固有サービス層１５０を直接サポートする。前記Ｆａｂｒｉｃ　ＣｏｕｃｈＤＢ（登録商標）機能２２０は、特定のインデックス作成機能に直接アクセスするために利用される。前記ブロックチェーンプラットフォーム１８０への依存を通じて、前記ブロックチェーンプラットフォーム固有サービス層１５０´は、（内部キャッシュサービスを通じて）ＣｏｕｃｈＤＢ（登録商標）２２０へのＧＥＴコールを行うことができるが、ＰＯＳＴコールまたは前記ブロックチェーン上のデータ操作は、前記ＡＳＩＫ　Ｔｒａｃｋチェーンコード１７０によってのみ実行される。異なるタイプのユーザーは、異なるタイプのデータ操作にアクセスできる。これらのユーザーは、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ認証局を使用して前記ＡＳＩＫ　Ｔｒａｃｋチェーンコードプラットフォーム１８０に登録することができる。また、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ上での前記ＡＳＩＫ　１００の実装は、前記ブロックチェーンプラットフォーム１８０へのスマートコントラクトのインストールを可能にするかもしれない。このようにして、ビジネスロジックは前記ブロックチェーンプラットフォーム固有Ｎｏｄｅ．ｊｓ（登録商標）Ｆａｂｒｉｃサービス層１５０´と前記ブロックチェーンプラットフォーム１８０自体の間で分割される。

【００７４】

ＡＳＩＫ　Ｔｒａｃｋサービスは、チェーンコード１７０が前記元帳１９０を更新するために必要な元帳とスマートコントラクトをホストするＦａｂｒｉｃピア接続２３０を通じて、前記ＡＳＩＫ　Ｔｒａｃｋチェーンコード１７０を呼び出す。Ｆａｂｒｉｃピア２３０は、前記ブロックチェーンネットワーク用の元帳とスマートコントラクトのインスタンスをホストする。前記台帳は、スマートコントラクト（Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃではチェーンコード１７０またはスマートコントラクトに含まれ、サポートされているプログラミング言語で記述された台帳にアクセスするコードの一部）によって生成されたすべてのトランザクションを不変的に記録する。スマートコントラクトと台帳はそれぞれ、ネットワーク内の共有プロセスと共有情報をカプセル化するために使用される。Ｆａｂｒｉｃピア２３０を通じて、アプリケーションはチェーンコードを実行し、台帳を照会または更新することができる。

【００７５】

前記ブロックチェーンプラットフォーム１８０への実際の呼び出しは前記ＡＳＩＫ　Ｔｒａｃｋチェーンコード１７０を通じて行われる。前記ＡＳＩＫ　Ｔｒａｃｋチェーンコード１７０が前記元帳１９０を更新するコールを行うと、そのレスポンス（成功／失敗）が前記ブロックチェーンプラットフォーム固有サービスレイヤ１５０または前記ＡＳＩＫウェブサービスＡＰＩ層１３０に送り返される。前記ＡＳＩＫ　Ｔｒａｃｋチェーンコード１７０を呼び出した後、前記ブロックチェーンプラットフォーム固有サービス層１５０はレスポンスを前記ＡＳＩＫウェブサービスＡＰＩ層１３０に送り返し、前記ＡＳＩＫウェブサービスＡＰＩ層１３０はレスポンスを前記クライアントアプリケーション１１０に送り返す。

【００７６】

ＡＳＩＫ　Ｔｒａｃｋ構成２００はマルチテナントシステムであるため、複数のレベルのアクセス制御を組み込んでいる。トランザクションデータに対する操作（読み取りまたは書き込み）を実行するために、呼び出し元はコアＡＳＩＫ参加者のＩＤと同様にテナントのＩＤを提供する。前記ＡＳＩＫ１００のボディの階層は、プラットフォーム管理者、テナント管理者、およびコアＡＳＩＫ参加者のために、以下のように表すことができる：

プラットフォーム管理者＞テナント管理者＞コアＡＳＩＫ参加者＞ＡＳＩＫ固有データ

【００７７】

図３は、サンプル構成におけるＡＳＩＫ　Ｔｒａｃｋアプリケーションの実装のためのＡＳＩＫ　Ｔｒａｃｋシステムオブジェクトの斜視図３００である。特に、図３は、前記ＡＳＩＫ　Ｔｒａｃｋ構成２００の様々なデータベースエンティティ間の関係を説明する。Ｆａｂｒｉｃ上の前記ＡＳＩＫ１００のデータ実現には、４つの役割が含まれる：プラ

ットフォーム管理者３０２、テナント管理者３０４、サプライチェーン管理者３０６、監査人／内部監査人３０８である。プラットフォーム管理者３０２は、製品展開時に登録される。この役割は、テナントを前記システム３００に登録する責任を負う。テナントはＩＤとシークレットを持ち、これを使用してルールと機能の作成／読み取り／更新／削除を行うことができる。テナントは、アプリケーション１１０を作成する責任も負う。アプリケーション１１０は、製品のＡＰＩレイヤでルール実行要求を開始するために使用される、独自のＩＤとシークレットを持つ。トランザクションのために一緒に実行されることを意図した複数のルールをグループ化することができる。

【００７８】

図３では、接続リンクの両端の数字は、エンティティ間の関係を番号で表している。例えば、図３では、参加者３１０は「１」と表示され、移動可能製品３２０は「１＊」と表示されているが、これは、１つの移動可能製品３２０は１つの参加者３２０にのみ関連することができるが、１つの参加者３１０は任意の数の移動可能製品３２０に関連することができることを意味する。図３は、例えばＡＳＩＫ　Ｔｒａｃｋ：

プラットフォーム管理者＞テナント管理者＞サプライチェーン管理者＞ＡＳＩＫ固有データ

プラットフォーム管理者＞テナント管理者＞サプライチェーン管理者＞監査人／内部監査人＞ＡＳＩＫ固有データ

プラットフォーム管理者＞テナント管理者＞サプライチェーン管理者＞ユーザー＞ＡＳＩＫのユーザーまたはユーザーグループ関連のトランザクションデータ

【００７９】

前記ＡＳＩＫのＨｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃへの実装は、前記ブロックチェーンプラットフォーム１８０へのスマートコントラクト／チェーンコードのインストールを可能にする。ＡＳＩＣＫチェーンコード１７０はＡＳＩＫのコア機能を含む。前記ＡＳＩＫ　Ｔｒａｃｋ構成では、前記ＡＳＩＣＫチェーンコード１７０は、参加者３１０、移動可能な製品３２０、製品３３０、ワークフロー３４０、ユーザーグループ３５０、サプライチェーン管理者３０６、トラッカー３６０などの作成を含む、あらゆる種類の取引データをＨｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ上に作成する機能を持つ。

【００８０】

前記ブロックチェーンプラットフォーム固有サービス層１５０´は、前記ＡＳＩＫウェブサービスＡＰＩ層１３０からの入力を受け入れ、前記ＡＳＩＣＫチェーンコード１７０およびＦａｂｒｉｃ　ＣｏｕｃｈＤＢ（登録商標）世界状態データベース２２０を呼び出すことによって、入力をアクションに変換する。前記ブロックチェーンプラットフォーム固有サービス層１５０´は、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ認証局と対話し、前記ＡＳＩＫ　１００に関与するすべての役割の証明書を生成する。ＡＳＩＫ　Ｔｒａｃｋ２００の場合、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ認証局はユーザー（参加者、サプライチェーン管理者、監査人、内部監査人）を登録し、証明書を生成する。これらの証明書は、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃネットワーク上でユーザーを検証するために、前記ＡＳＩＫウェブサービスＡＰＩ層１３０を使用して、そのユーザーが前記ＡＳＩＫ　１００上でリクエストを行う際に使用される。

【００８１】

ＡＳＩＫテナントオンボードサービス１２０は、テナント作成中に様々なフェーズを実装する。例えば、ファブリックリソース割り当てフェーズでは、（クライアント／ユーザーの要求に応じて）専用または共有リソースを割り当ててもよく、前記ＡＳＩＫ　１００が前記ブロックチェーンネットワーク用のＢＮＯサービスに使用する要求に従ってリソースをオンザフライで作成してもよい。チャネルの作成とスマートコントラクトの展開フェーズでは、テナント専用のチャネルが作成され、前記チェーンコード１７０がそのチャネルにインストールされ、初期化されてもよい。サンプル構成では、前記チャネルは、機密トランザクションの実施に使用するためのネットワークの２若しくはそれ以上のノード間

の通信チャネルである。ＣｏｕｃｈＤＢ（登録商標）ビュー作成フェーズでは、スマートコントラクトまたはチェーンコード１７０を展開した後、ＣｏｕｃｈＤＢ（登録商標）ノード内の専用データベースを作成してもよい。データ検索プロセスのために、ビュー文書もテナント固有のＣｏｕｃｈＤＢ（登録商標）２２０に作成されてもよい。Ｋａｆｋａトピックパーティション作成フェーズでは、より高速な処理を保証するために、テナント専用の構成可能な数のパーティションが追加されてもよい。最後に、Ｅｌａｓｔｉｃｓｅａｒｃｈインデックス作成フェーズでは、前記ブロックチェーンプラットフォーム固有サービス層１５０は、より高速なデータ検索のためにテナント固有のインデックスを作成するためにＥｌａｓｔｉｃｓｅａｒｃｈを使用してもよい。

【００８２】

前記ＡＳＩＫウェブサービスＡＰＩ層１３０は、クライアントアプリケーション１１０が使用するＡＰＩを公開している。前記ＡＳＩＫ　Ｔｒａｃｋの設定では、ＡＳＩＫ　Ｔｒａｃｋの機能をカバーする、前記クライアントアプリケーション１１０に公開されるＡＰＩのセットがある。ＡＳＩＫ　ＴｒａｃｋのサンプルＡＰＩを以下の表１に示す：

【００８３】

【表１】

表１

| タグ | ＡＰＩ／機能 |
|---|---|
| サプライチェーン管理者 | サプライチェーン管理者を作成／取得 |
| ユーザー | ユーザーを作成／取得 |
| ユーザーグループ | ユーザーグループを作成／取得 |
| 監査人 | 監査人を作成／取得 |
| 内部監査人 | 内部監査人を作成／取得 |
| コンテキスト変数 | コンテキスト変数を取得 |
| 争議 | 争議を作成／取得／解決 |
| 製品 | 製品を作成／取得 |
| ワークフロー | ワークフローを作成／更新／編集／取得 |
| | 可動製品を属性名およびその他のろ過で取得 |
| | 可視性設定に基づく可動製品数の取得 |
| 可動製品 | 可動製品の作成／更新／転送／受領／返却 |
| | 最後の機能ノードを実行 |
| | ノード実行ログの取得 |
| トラッカー | 可動製品ＩＤによるトラッカー文書の取得 |
| | 可動製品ＩＤと属性名によるトラッカー文書の取得 |

【００８４】

表１は、異なるタイプのタグと、それらに関連するＡＰＩまたは機能を示している。これらのＲＥＳＴ　ＡＰＩはクライアントアプリケーションに公開され、前記クライアントアプリケーション１１０の要件を満たすために固有の機能を実行する責任を負う。例えば、タグ「ＵＳＥＲ」は「Ｃｒｅａｔｅ　Ｕｓｅｒ」ＡＰＩを持ち、前記クライアントアプリケーション１１０にユーザを作成する方法を提供し、そのユーザは移動可能製品を作成／転送／受け取ることができる。

【００８５】

前記ブロックチェーンプラットフォーム固有サービス層１５０は、リクエストレスポンスサイクルをカバーすることができる。表１は、サンプル構成におけるＡＳＩＫ　Ｔｒａｃｋの機能を示している。表１の各タグに対して個別のサービスが作成されてもよい。この層の構造を図４に例として示す。

【００８６】
図４は、ＡＳＩＫ　Ｔｒａｃｋのサンプル構成の制御フローを示すフロー図４００である。図４は、ＡＳＩＫ　Ｔｒａｃｋの異なる構成要素の相互作用を示し、クライアントアプリケーション１１０から、またクライアントアプリケーション１１０に戻る制御フローを完成させる。前記クライアントアプリケーション１１０はＡＳＩＫ　Ｔｒａｃｋ２００に接続し、前記ＡＳＩＫ　Ｔｒａｃｋ２００はクライアントコネクタ４１５を介してＲｅｄｉｓ、ＰｏｓｔｇｒｅＳＱＬ、Ｅｌａｓｔｉｃｓｅａｒｃｈなどの外部サービス４１０に接続し、所望の機能を実現する。
【００８７】
前記ＡＳＩＫウェブサービスＡＰＩ層１３０は、前記クライアントアプリケーション１１０から直接コマンドを受け取る。前記ＡＳＩＫウェブサービスＡＰＩ層１３０は、リクエスト処理後に前記クライアントアプリケーション１１０に前記レスポンス（成功または失敗）を返す層でもある（４２５）。前記ＡＳＩＫウェブサービスＡＰＩ層１３０は、入力の秘密にするべき情報を除去するためにリクエストをＡＳＩＫ　Ｔｒａｃｋミドルウェア４２０に転送する。前記ＡＳＩＫ　Ｔｒａｃｋミドルウェア４２０は、前記ＡＳＩＫウェブサービスＡＰＩ層１３０からの前記入力を検証し、前記リクエストが事前に定義されたスキーマに準拠していることを確認する。また、前記ＡＳＩＫ　Ｔｒａｃｋミドルウェア４２０は入力の秘密にするべき情報を除去し、前記ブロックチェーンプラットフォーム固有サービス層１５０が前記入力を処理するために必要なフォーマットに変換する（４３０）。どの時点でも、前記リクエストが事前に定義されたスキーマにマッチしない場合、前記リクエストは拒否され、前記ＡＳＩＫウェブサービスＡＰＩ層１３０に戻される(４３５)。検証後、前記リクエストは前記ブロックチェーンプラットフォーム固有サービス層１５０に転送される。
【００８８】
サンプル構成では、前記ＡＳＩＫ　Ｔｒａｃｋサービス層１５０は１若しくはそれ以上のＡＳＩＫ　Ｔｒａｃｋサービスのグループである。各サービスは、それに割り当てられた機能を実行し、次の／前のサービスに応答を返す責任を負う。Ｒｅｄｉｓ、Ｋａｆｋａ、ＰｏｓｔｇｒｅＳＱＬ、Ｅｌａｓｔｉｃｓｅａｒｃｈ、ＣｏｕｃｈＤＢ（登録商標）などの外部サービス４１０へのクライアント接続またはコネクタ４１５も前記ＡＳＩＫ　Ｔｒａｃｋサービス層１５０に確立される。クライアント接続またはコネクタ４１５は、前記ＡＳＩＫ　Ｔｒａｃｋサービス層１５０が前記外部サービス４１０（４４０）と相互作用する層を提供する。前記ＡＳＩＫ　Ｔｒａｃｋサービス層１５０はＡＳＩＫ　Ｔｒａｃｋの各機能部分の処理を担当する。ＡＳＩＫ　Ｔｒａｃｋの各機能部分の処理を担当するために、個別のサービスファイルが作成される（４４５）。前記ＡＳＩＫ　Ｔｒａｃｋサービス層１５０は受け取った要求に基づいて次のサービスを呼び出すことを決定する。外部サービス４４０から受け取ったデータは、それぞれのサービスに戻すことができる（４５０）。前記ブロックチェーンプラットフォーム固有サービス層１５０´において、前記ＡＳＩＫ　Ｔｒａｃｋサービスは前記Ｈｙｐｅｒ　Ｆａｂｒｉｃネットワークと通信してもよい（４５５）。
【００８９】
ＡＳＩＫ　Ｔｒａｃｋのサンプル構成では、事前に設定されたシナリオとビジネスルールが、結果を安全にするためにブロックチェーン上で同じことをコミットした後、累積結果を提供するために、１つの属性のトランザクションを集約するために使用される。ＡＳＩＫ　Ｔｒａｃｋの構成では、ビジネス機能のためにいくつかのノードが追加される。各ノードには、ＡＳＩＫ　Ｔｒａｃｋに関連するいくつかの操作を実行するために、ＪａｖａＳｃｒｉｐｔ関数（スクリプトノード、トランジションノード、ワークフロートリガーノード）またはサードパーティＡＰＩ実行（サービスノード）の形でクライアントアプリケーション１１０から与えられる一連の命令が含まれる。一例では、前記ノードは、スクリプトノード、サービスノード、トランジションノード、およびワークフロートリガーノードを含むことができる。

【００９０】
スクリプトノードは、前記ワークフローの最初のノードにすることはできない。宣言の例を以下に示す：{{foo}}={{bar}}+10;}。２若しくはそれ以上のスクリプトノードを連続させることができる。前記スクリプトノードは、ユーザーグループノードマッピングには属さないが、ノードＩＤ転送シーケンスには属することがある。インスタンス移動時に、インスタンスが前記スクリプトノードを通るパスをたどると、前記スクリプトノードに記述された前記スクリプトが実行される。前記スクリプトノードに設定された属性値は、属性で設定され、スクリプトノードの次のグループノードで可視化される場合がある。スクリプト／サービスノードで何らかの障害が発生した場合、前記インスタンスは、前のグループノードに残る。例えば、ワークフローの場合：ユーザーグループノード１＞スクリプトノード１＞ユーザーグループノード２の場合、前記スクリプトノードでＡＢＣの属性値が１２３に設定されると、ユーザーグループノード２にはＡＢＣの属性値が１２３として表示される。
【００９１】
また、サービスノードも前記ワークフローの最初のノードにすることはできない。２若しくはそれ以上のサービスノードを連続させることができる。前記サービスノードは、前記ユーザグループノードマッピングに属さないが、前記ノードＩＤ転送シーケンスに属することがある。インスタンス移動時に、前記インスタンスが、前記サービスノードを経由する経路をたどる場合、前記サービスノードは、サードパーティＡＰＩを呼び出して実行されることがある。前記サービスノードは、設定されたタイムアウトまで前記サードパーティＡＰＩからの応答を待つ。前記サービスノードは、設定されたリトライ値時間に従ってリトライを試みることができる。
【００９２】
トランジションノードも最初のノードになることはできない。前記トランジションノードは、前記ユーザーグループノードマッピングには属さないが、前記ノードＩＤ転送シーケンスには属することがある。インスタンスの移動時に、前記インスタンスが、前記スクリプトノードを通るパスをたどる場合、前記トランジションノードで言及された前記スクリプトが実行され、前記トランジションノードは「to_node_id」を返す。
【００９３】
ワークフロートリガーノードの、前記ワークフローの最初のノードになることができない。また、前記ワークフロートリガーノードは、前記ユーザグループノードマッピングには属さないが、前記ノードＩＤ転送シーケンスには属することができる。インスタンス移動時に、前記インスタンスが、前記ワークフロートリガーノードを通る経路をたどる場合、バッチコンポーザサービス（ＢＣＳ）が呼び出されてバッチが生成され、生成されたバッチＩＤが、前記クライアントアプリケーション１１０に返される。例えば、以下はワークフロートリガーノードの宣言例である：
【００９４】
【数１】

```
"aggregation_settings": {
   "aggregation": {
      "script": "{{DL.AGG.var1}} = AGGR(\"SUM\", \"{{foo}}\");"
   },
   "post_aggregation": {
      "script": "{ \n    {{DL.WFINSTANCE.wf1}} = {{DL.AGG.var1}} +150;\n     {{foo}} =
{{DL.AGG.a1}} + 500;\n}"
   },
   "condition": {
      "script": "function(){ \n    if ( {{bar}} == 1001 && {{foo}} > 10 ) { \n return true; \n    } e
lse { \n       return false; \n    } \n}"
   },
   "scheduler": {
       "scheduler_type": "DAILY",
       "scheduler_execution_time": {
          "execution_time": "18:13:00",
          "time_zone": "UTC+05:30"
       }
    }
}
```


【００９５】
サンプル構成では、以下のサービスがＡＳＩＫ　Ｔｒａｃｋアプリケーションに含まれる。

【００９６】
転送、受信、更新、最終ノード実行など、前記可動製品関連トランザクションのバッチを生成する役割を担うバッチコンポーザサービス（ＢＣＳ）が提供される場合がある。前記ＢＣＳは、生成されたバッチＩＤを前記ＡＳＩＫ　Ｔｒａｃｋサービスに返し、生成されたバッチＩＤをＫａｆｋａキューに送り集計を行う。クライアントアプリケーション１１０によって与えられた集約設定の条件が偽であるまで、前記トランザクションは同じバッチの一部である可能性がある。条件が真になると、前記バッチは完了する。

【００９７】
アグリゲーションエンジンサービスは、アグリゲーションおよびポストアグリゲーションプロセスの実行を担う。前記アグリゲーションは、可動製品属性値に対して実行される場合がある。例えば、ＳＵＭ、ＭＩＮ、ＭＡＸ、ＣＯＵＮＴ、およびＡＶＧ集計関数がサポートされる場合がある。前記トランザクション／要求が受け取られると同時に実行される実行時集計であるＥａｇｅｒ　Ａｇｇｒｅｇａｔｉｏｎと、前記バッチが完了すると前記アグリゲーションを実行するＬａｚｙ　Ａｇｇｒｅｇａｔｉｏｎの２種類のアグリゲーションが実装される場合がある。ポストアグリゲーションは、前記アグリゲーションが完了した後に実行される。ポストアグリゲーションは通常、ワークフロートリガーノードで定義されたスクリプトに従って属性値を変更するために使用される。

【００９８】
前記ワークフロートリガーノードの後、前記定義されたワークフローの次のフローを完了させる役割を担う可動製品ワーカーサービスが提供される場合がある。例えば、Ｗｏｒｋｆｌｏｗ１の例の場合：ユーザーグループノード１＞ワークフロートリガーノード１＞ユーザーグループノード２の場合、アグリゲーションとポストアグリゲーションが完了すると、Ｋａｆｋａキュー経由で可動製品ワークサービスにリクエストが来る。前記可動製

品ワークサービスは、次のノードがグループノードであることを確認し、ユーザーグループノード２で転送機能を実行する。一方、Ｗｏｒｋｆｌｏｗ２の例：ユーザーグループノード１＞ワークフロートリガーノード１＞ワークフロートリガーノード２＞ユーザーグループノード２＞ｗｔｎ１＞ｗｔｎ２＞グループ２の場合、前記ワークフロートリガーノード１のアグリゲーションとポストアグリゲーションが完了すると、Ｋａｆｋａキュー経由で可動製品ワーカーサービスにリクエストが来る。前記可動製品ワーカーサービスは、次のノードがワークフロートリガーノードであるかどうかをチェックする。もしそうなら、前記可動製品ワーカーサービスは、ワークフロートリガーノード２の処理のために前記バッチコンポーザーサービスを呼び出す。前記ワークフロートリガーノード２の処理が完了すると、前記可動製品ワーカーサービスは、ユーザーグループノード２に対して転送機能を実行することができる。

【００９９】

可動製品トレースデータハンドラサービスは、前記可動製品のトレース詳細を管理するために使用される。前記可動製品トレースデータハンドラサービスは、定義されたビジネスルールに基づいて、トレース情報を永続環境（Ｐｏｓｔｇｒｅｓ）にロードするバックグラウンドジョブを持つ。また、前記可動商品トレースデータハンドラサービスは、前記提供された可動商品のトレース詳細を閲覧するためのＡＰＩを持つ。前記可動製品トレースデータハンドラサービスを利用することで、ユーザ／クライアントは、可動製品ＩＤを持つ全てのキー属性（または）を集約したインスタンスのリストをトレースすることができる。

【０１００】

前記ワークフロートリガーノードのスケジューリングジョブを作成するスケジューリングコンポーザサービスが提供されてもよい。前記スケジューリングジョブの設定は、前記クライアントアプリケーション１１０がワークフローを作成する際に、以下のように入力する：

【０１０１】

【数２】

```
"scheduler": {
        "scheduler_type": "DAILY",
        "scheduler_execution_time": {
            "execution_time": "18:13:00",
            "time_zone": "UTC+05:30"
        }
}
```

【０１０２】

このスケジューリングジョブは、保存されたワークフローの詳細と前記バッチＩＤの集計を実行するリクエストをトリガーする可能性がある。前記スケジューラコンポーザサービスは、前記クライアントアプリケーション１１０が、スケジューラ設定を持つワークフローを作成する際に、リクエストを受け取ることがある。

【０１０３】

上述したように、前記ブロックチェーンプラットフォーム固有サービス層１５０´は、前記Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ（ブロックチェーン）ネットワークとの通信を行う。前記ブロックチェーンプラットフォーム固有サービス層１５０´は、主要なビジネスロジックを保持し、データ操作を管理し、前記ブロックチェーンプラットフォーム１８０を呼び出す。前記ブロックチェーンプラットフォーム固有サービス層１５０´は、前記ＡＳＩＫトラックサービス層１５０からの入力を受け入れ、前記ＡＳＩＫチ

ェーンコード１７０を呼び出すフォーマットで前記リクエストを作成する。前記ＡＳＩＫチェーンコード１７０は、前記ＡＳＩＫ　１００が、前記ブロックチェーンプラットフォーム１８０に前記リクエストを提出し、前記元帳と状態データベース（例えば、ＣｏｕｃｈＤＢ（登録商標））にデータを入れる層である。前記ブロックチェーンプラットフォーム固有サービス層１５０´はまた、前記Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ認証局と相互作用して、ＡＳＩＫ　１００に関わるすべての役割の証明書を生成する。ＡＳＩＫ　Ｔｒａｃｋ構成の場合、前記Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃ認証局はユーザー（参加者、サプライチェーン管理者、監査人、内部監査人）を登録し、証明書を生成することができる。これらの証明書は、そのユーザーがＡＳＩＫウェブサービスＡＰＩ層１３０を使用して前記ＡＳＩＫ　１００上であらゆるリクエストを行う際に使用され、前記ブロックチェーンプラットフォーム１８０上でユーザーを検証することができる。

【０１０４】

以下の表２は、様々な機能やビジネス要件に対応するためにＡＳＩＫ　Ｔｒａｃｋで作成される様々なマイクロサービスのサンプルリストである。

【０１０５】

【表２】

表２

| サービス | 目的 |
|---|---|
| ワークフローとユーザー管理サービス | このサービスは、ワークフローとユーザーに関するすべてのトランザクションを担当する。 |
| 可動製品サービス | このサービスは、可動製品に関連するすべての取引を担当する。 |
| バッチ作成者サービス | このサービスは、動産トランザクションのバッチを生成し、集計が実行されるように先に送信する責任を負う。 |
| 可動製品ワーカーサービス | このサービスは、定義されたワークフローの次のフローを完了するために集約されたすべてのトランザクションを担当する。 |
| 可動製品データハンドラーとトラッキングサービス | このサービスは、可動製品のスナップショットとその属性のトレースを保存する役割を果たす。 |
| スケジューリング作成者サービス | このサービスは、定義された時間間隔でトリガーされるcronジョブのスケジューリングを担当する。このジョブは保存されたワークフロー詳細の集計を実行するリクエストをトリガーする。 |
| 発見とリスナーサービス | このサービスは、データがElasticsearchに同期されるようにソース抽出サービスのインスタンスを割り当てる役割を担っている。 |
| リソース管理サービス | このサービスは、テナントのファブリック・リソースの割り当てをすべて担当する。 |
| テナント管理サービス | このサービスは、システム内にテナントを作成し、同じテナントのリソースを割り当てる。 |
| アグリゲーションエンジンサービス | このサービスは、ワークフ ロー・トリガー・ノードで定義されたスクリプトに従って、可動製品属性値を集計し、 属性に値を割り当てる。 |
| ソース抽出サービス | このサービスは、ブロックマイニング成功時のイベントとしてHyperledger（登録商標）Fabricピアからドキュメントを抽出し、トランスフォーマーとローダーサービスに送信する役割を担っている。 |
| トランスフォーマーとローダーサービス | このサービスは、Hyperledger（登録商標）Fabricピアから抽出されたドキュメントをElasticsearchに保存し、データをより速く検索する役割を担っている。 |

【０１０６】

表２に記載された各サービスは、前記ＡＳＩＫプラットフォームがインターフェースされる前記クライアントアプリケーション１１０に公開される複数のＡＰＩを含む可能性がある。例えば、サービス「可動製品トレースデータハンドラおよびトラッキングサービス」は、前記可動製品のスナップショットとその属性のトレースの保存を担当する。

【０１０７】

図５は、一般的な構成のＡＳＩＫ　１００の動作を示すフロー図５００である。

【０１０８】

図５に示されるように、前記ＡＳＩＫ　１００の開始点は、分散型台帳製品のフロントエンドを扱う企業レベルのユーザーである。工程は、５１０でのユーザーの要求または入力から始まる。ユーザーによって提供されるデータは、前記ブロックチェーンネットワーク上で何らかのトランザクション１８５を実行する要求である。このリクエストは、ユーザーインターフェース５２０から、ＨＴＴＰＳリクエストによって、またはＲｅｓｔｆｕ

ｌ　ＡＰＩ１４０から行われるリクエストによって、前記ユーザーによって行うことができる。

【０１０９】

通常、前記リクエストは、前記分散台帳製品のフロントエンドのレスポンシブユーザーインターフェイス５２０、ＨＴＴＰＳリクエスト、またはＲＥＳＴｆｕｌ　ＡＰＩ１４０から転送される。これらのリクエストは、関数[f(x)Webservice]の形でエンドポイント（ＡＰＩ）から前記ＡＳＩＫウェブサービスレイヤー１５０にルーティングされる。前記リクエストの性質は、前記ＡＳＩＫウェブサービスレイヤー１５０で呼び出される必要な関数を決定し、この工程で特定されるビジネスロジックを導く。

【０１１０】

前記パッケージが、ＡＳＩＫウェブサービス層１５０にルーティングされる前に、前記リクエストは、ミドルウェア４２０によって検証されるかもしれない。前記ミドルウェア４２０は、前記提供されたリクエストがビジネス要件に従って全てのパラメータを持つかどうかをチェックする。前記ミドルウェア４２０は、例えば、パラメータや入力変数名の大文字小文字の区別や、そのような類似の機能など、特定の特別な機能も処理する。

【０１１１】

ビジネス要件に応じて、前記ＡＳＩＫ　１００のウェブサービス機能[f(x)Webservice]によって、これらの要件に対応する適切な機能が選択される。リクエスト関数は、次に前記チェーンコード１７０の同様の関数[f(x)Chaincode]をトリガーする。これらの関数は、入力されたリクエストを、前記チェーンコード１７０を介して前記ブロックチェーンプラットフォーム１８０上で実行可能なトランザクションに変換する変調器として機能する。したがって、すべてのリクエスト関数に対して、前記ブロックチェーンプラットフォーム１８０によって解釈されるフォーマットで情報を変換する同様のレスポンス関数が存在する。ビジネス上の問題はチェーンコード層１７０で解決され、前記チェーンコード層１７０は前記ブロックチェーンプラットフォーム１８０とのすべてのやり取りを担当する。

【０１１２】

前記トランザクション１８５が前記ブロックチェーンプラットフォーム１８０上で実行されると、業務が完了する。その後、トランザクション１８５の状態は、企業によって行われた全てのトランザクション１８５を追跡する事前設定された状態データベース（例えば、ＣｏｕｃｈＤＢ（登録商標））２２０上に更新される。前記ＡＳＩＫ　１００内のすべてのトランザクションは、内部および外部のトランザクション１８５が安全なチャネルを介して送信されるように、セキュリティプロトコル（例えば、トランスポートレイヤーセキュリティ（ＴＬＳ）１．０）に従って実行される場合がある。

【０１１３】

トランザクション１８５が、前記ブロックチェーンプラットフォーム１８０上で実行されるたびに、前記台帳１９０を更新する工程が順次行われる。前記トランザクションのステータス、すなわち成功（５３０）または失敗（５４０）にかかわらず、一意のＩＤを持つトランザクション１８５が前記台帳１９０上で更新される。前記スマートコントラクト／チェーンコード１７０は、取引報告をさらに強化するために前記ＡＳＩＫ　１００によってカスタマイズされてもよい。

【０１１４】

トランザクション１８５が失敗するたびに（５４０）、技術的エラー再処理５５０のようなエラー処理システムは、前記トランザクション１８５を前記ＡＳＩＫ　１００に割り当てる。前記ＡＳＩＫ　１００は、エラーの性質を識別し、前記トランザクション１８５を適切な開始点から再実行する。

【０１１５】

ビジネス操作後、前記リクエストは完了し、前記レスポンスは、前記チェーンコード１７０を介して前記ＡＳＩＫウェブサービス層１５０に伝搬する。前記レスポンスは、その後、前記ユーザーインターフェース５２０上のユーザーインターフェースダッシュボード

を介して望ましい出力として表示されることができる。このように、前記ＡＳＩＫウェブサービス層１５０を起点とするリクエスト機能のライフサイクルは、レスポンスの形で同じ位置で終了する。

【０１１６】

当業者であれば、Ｆａｂｒｉｃ／Ｎｏｄｅ．ｊｓ（登録商標）で実装される前記ＡＳＩＫ　１００が、分散化、安全なデータ暗号化、実装の容易さ、および高い不正防止という利点を有することを理解するであろう。前記ＡＳＩＫ　１００は、ブロックチェーンの機能や内容を中断することなく、ブロックチェーンの中核機能を保持し、ユーザーに伝える。前記ＡＳＩＫ　１００はまた、事前に設定されたシナリオを備えた「プラグ・アンド・プレイ」ソリューションを提供する。ユーザーは、従来のユーザーインターフェースの方法またはＨＴＴＰＳリクエストを使って前記ＡＳＩＫウェブサービス層１５０にリクエストを送り、前記ＡＳＩＫ　１００が残りを処理する。ユーザーは、前記ＡＳＩＫ　１００を使用するために設定する要素は最小限である。ビジネスロジックは正確に構築され、チャネル内のピアに展開され、ビジネスロジックは前記ブロックチェーン上で実行される。モジュール化された性質により、ブロックチェーンの技術的理解の必要性を抽象化した特定のユースケースに適応したブロックチェーン上のトランザクションの実行操作が簡素化され、それによって前記ブロックチェーンアプリケーションはむしろ使いやすくなる。また、最も求められているビジネスユーティリティと機能は、プログラミングインターフェースを介してルールで再利用可能なものとして事前に構成されており、企業が様々な複雑性に同じソリューションを使用できることを保証する。米国特許出願第１７／６５３，０８５号に記載されている前記ブロックチェーンルールエンジンも、前記ＡＳＩＫ　１００によってビジネスルールを実施するために使用されるだけでなく、そのためにユーザーアプリケーションに入ることなく新しいルールの作成を可能にするために使用される可能性があることがさらに理解されるであろう。米国特許出願第１７／６５３，０８５号の内容は、参照によりその全体がここに組み込まれる。

【０１１７】

前記ＡＳＩＫ　１００はまた、ユーザーと前記ブロックチェーンの間に信頼できる後方互換性のあるインターフェースを提供する。前記ＡＳＩＫ　１００は、前記ブロックチェーン上のすべての操作を管理するため、ユーザーは前記ブロックチェーンを理解しなくても前記ＡＳＩＫ　１００のメリットを享受できる。また、前記ＡＳＩＫ　１００は、特定のビジネス要件を満たすために公開されるＡＰＩの集合体を提供するため、前記ＡＳＩＫ１００は、ＡＰＩの互換性と非常によく似ている任意の技術スタックと連携して使用することができます。前記ＡＳＩＫ　１００は、また、チェーンコードとスマートコントラクトの両方で動作し、Ｅｔｈｅｒｅｕｍ、Ｈｙｐｅｒｌｅｄｇｅｒ（登録商標）Ｆａｂｒｉｃなどの複数のブロックチェーンネットワークで動作するため、ブロックチェーンにとらわれない。前記ＡＳＩＫ　１００はまた、すべての主要な技術スタックとプログラミング言語にわたって動作し、負荷分散されたＲＥＳＴ　ＡＰＩを介して互換性のあるさまざまなブロックチェーンやクラウドネットワークと相互作用する。

【０１１８】

前記ＡＳＩＫ　１００は、すべての主要な技術スタックとプログラミング言語にわたって動作することを可能にするセマンティクス、構文、インターフェイス、要件を概説しているため、ビジネスサービス層は異なる技術で開発することができ、新しい開発は最新の技術フレームワークで行い、技術的な利点を活用することができる。前記ＡＳＩＫ　１００は、非ブロックチェーンプログラマにとって既に馴染みのあるセマンティクスと構文を提供し、すべてのブロックチェーンプラットフォームで同一であるため、迅速なアプリケーション開発が可能である。前記ＡＳＩＫ　１００はさらに、設定を変更するだけで複数のクライアントアプリケーションと連携できるプラグイン可能なプラットフォームを提供する。負荷分散された強力なＲＥＳＴ　ＡＰＩを使えば、さまざまなブロックチェーンフレームワークと比較的簡単にやり取りできる。その結果、フレームワークは、前記ブロックチェーンのアップグレードやリリースを容易にする。また、前記ブロックチェーンにマ

ルチテナントを実装することで、複数のテナントが前記ブロックチェーン上でデータを作成・管理し、それぞれのテナントのデータを物理的に分離することが可能になる。テナントとＡＳＩＫのコア参加者に関するすべての操作は、マルチテナントの枠組みの中で前記ブロックチェーン上に安全に記録される。最後に、あらかじめ設定された機能と特徴のモジュール式フレームワークにより、前記ＡＳＩＫ　１００は、複数のビジネスユースケースを容易に実現するプラットフォームを提供することができる。

【０１１９】

これらおよびその他の利点は、上記の説明から当業者には明らかであろう。

【０１２０】

コンピューターの実施形態

図６は、本明細書で開示する前記ＡＳＩＫ　１００の１若しくはそれ以上の実施形態を実装するのに適した特殊用途コンピューターにプログラムすることができる、典型的な汎用コンピュータ６００のブロック図である。上述した前記ＡＳＩＫ　１００は、必要な作業負荷を処理するのに十分な処理能力、メモリ資源、および通信スループット能力を有するコンピュータなどの任意の汎用処理構成要素上に実装することができる。前記コンピューター６００は、二次記憶装置６０４、読み出し専用メモリ（ＲＯＭ）６０６、ランダムアクセスメモリ（ＲＡＭ）６０８、入出力（Ｉ／Ｏ）装置６１０、およびネットワーク接続装置６１２を含むメモリ装置と通信するプロセッサ－６０２（中央プロセッサ－装置またはＣＰＵと呼ばれることがある）を含む。前記プロセッサ６０２は、１若しくはそれ以上のＣＰＵチップとして実装されてもよいし、１若しくはそれ以上の特定用途向け集積回路（ＡＳＩＣ）の一部であってもよい。

【０１２１】

前記二次記憶装置６０４は、通常、１若しくはそれ以上のディスクドライブまたはテープドライブで構成され、データの不揮発性記憶のため、およびＲＡＭ６０８がすべての作業データを保持するのに十分な大きさがない場合のオーバーフローデータ記憶装置として使用される。前記二次記憶装置６０４は、ＲＡＭ６０８にロードされるプログラムが実行のために選択された場合に、そのプログラムを記憶するために使用することができる。前記ＲＯＭ６０６は、プログラムの実行中に読み出される命令およびおそらくデータを格納するために使用される。ＲＯＭ６０６は、不揮発性メモリ装置であり、通常、前記二次記憶装置６０４の大きなメモリ容量に対して小さなメモリ容量を有する。前記ＲＡＭ６０８は、揮発性データを記憶し、おそらく命令を記憶するために使用される。ＲＯＭ６０６とＲＡＭ６０８の両方へのアクセスは、通常、前記二次記憶装置６０４へのアクセスよりも高速である。

【０１２２】

本明細書で説明する装置は、コンピューター可読命令を記憶したコンピューター可読非一過性媒体と、前記メモリに結合された１若しくはそれ以上のプロセッサーとを含み、前記コンピューター可読命令を実行すると、図１から図５を参照して上述した方法工程および動作を実行するように前記コンピューター６００を構成することができる。前記コンピュータ読み取り可能な非一過性媒体には、磁気記憶媒体、光学記憶媒体、フラッシュ媒体、および固体記憶媒体を含む、あらゆるタイプのコンピュータ読み取り可能媒体が含まれる。

【０１２３】

本開示の工程のいずれか１つまたはすべてを参照して上述したような処理および動作を容易にする１若しくはそれ以上のコンピューター実行可能命令を含むソフトウェアは、さらに、本開示に一貫した消費者および／または生産者ドメイン内の１若しくはそれ以上のサーバーおよび／または１若しくはそれ以上のルータおよび／または１若しくはそれ以上の装置にインストールされ、それらとともに販売されてもよいことが理解されるべきである。あるいは、前記ソフトウェアは、例えば、ソフトウェア作成者が所有するサーバーから、または所有されていないが前記ソフトウェア作成者が使用するサーバーからを含む、物理的媒体または配布システムを通じて入手することを含め、本開示に一貫した消費者お

よび／または生産者ドメイン内の１若しくはそれ以上のサーバーおよび／または１若しくはそれ以上のルータおよび／または１若しくはそれ以上の装置に入手され、ロードされてもよい。前記ソフトウェアは、例えば、インターネット上で配布するためにサーバーに格納することができる。

【０１２４】

また、当業者であれば、本開示は、本明細書に記載または図面に例示された構造の詳細および構成要素の配置にその適用が限定されないことが理解されるであろう。本明細書における実施形態は、他の実施形態が可能であり、様々な方法で実施または実施することが可能である。また、本明細書で使用される言い回しや用語は、説明のためのものであり、限定的なものと見なすべきではないことが理解されよう。本明細書における「含む（ｉｎｃｌｕｄｉｎｇ）」、「有する（ｃｏｍｐｒｉｓｉｎｇ）」、または「持つ（ｈａｖｉｎｇ）」、およびそれらのバリエーションの使用は、それ以降に列挙される項目およびその等価物、ならびに追加の項目を包含することを意味する。別段の限定がない限り、本明細書における「連結」、「結合」、「取り付け」、およびその変形という用語は、広義に使用され、直接的および間接的な連結、結合、および取り付けを包含する。さらに、用語「連結」および「結合」並びにそれらの変形は、物理的または機械的な連結または結合に限定されない。さらに、上、下、底、上などの用語は相対的なものであり、説明を助けるために採用されているが、限定するものではない。

【０１２５】

図示の実施形態に従って採用される例示の装置、システム、および方法の構成要素は、少なくとも部分的に、デジタル電子回路、アナログ電子回路、またはコンピューターハードウェア、ファームウェア、ソフトウェア、またはそれらの組み合わせで実装されてもよい。これらの構成要素は、例えば、プログラマブルプロセッサ、コンピューター、または複数のコンピューターなどのデータ処理装置による実行のため、またはその動作を制御するために、情報担体、または機械可読記憶装置において具体化されたコンピュータープログラム、プログラムコード、またはコンピューター命令などのコンピュータープログラム製品として実装することができる。

【０１２６】

コンピュータプログラムは、コンパイル言語やインタプリタ言語を含む、あらゆる形式のプログラミング言語で記述することができ、スタンドアロンプログラムとして、またはモジュール、コンポーネント、サブルーチン、またはコンピューティング環境での使用に適した他のユニットとしてなど、あらゆる形式で配備することができる。コンピューター－プログラムは、１つのコンピューター－上で実行されるように配備されてもよいし、１つのサイトの複数のコンピューター－上で実行されるように配備されてもよいし、複数のサイトに分散され、通信ネットワークによって相互接続されるように配備されてもよい。また、本明細書に記載された技術を達成するための機能プログラム、コード、およびコードセグメントは、当業者であるプログラマによって本開示の範囲内であると容易に解釈され得る。例示的な実施形態に関連する方法工程は、コンピュータープログラム、コード、または命令を実行する１若しくはそれ以上のプログラマブルプロセッサーによって実行され、（例えば、入力データに対して動作すること、および／または出力を生成することによって）機能を実行することができる。方法工程はまた、例えば、ＦＰＧＡ（フィールドプログラマブルゲートアレイ）またはＡＳＩＣ（特定用途向け集積回路）などの特殊用途の論理回路によって実行されてもよく、装置は、特殊用途の論理回路として実装されてもよい。

【０１２７】

本明細書で開示される実施形態に関連して説明される様々な例示的論理ブロック、モジュール、および回路は、汎用プロセッサー、デジタル信号プロセッサー（ＤＳＰ）、ＡＳＩＣ、ＦＰＧＡまたは他のプログラマブル論理装置、ディスクリートゲートまたはトランジスタ論理、ディスクリートハードウェア構成要素、または本明細書で説明される機能を実行するように設計されたそれらの任意の組み合わせで実装または実行することができる。汎用プロセッサーは、マイクロプロセッサーであってもよいが、代替として、プロセッ

サーは、任意の従来のプロセッサー、コントローラ、マイクロコントローラ、またはステートマシンであってもよい。プロセッサーはまた、コンピューター装置の組み合わせ、例えば、ＤＳＰとマイクロプロセッサーの組み合わせ、複数のマイクロプロセッサー、ＤＳＰコアと組み合わせた１若しくはそれ以上のマイクロプロセッサー、または任意の他のそのような構成として実装されてもよい。

【０１２８】

コンピュータープログラムの実行に適したプロセッサーには、一例として、汎用および特殊用途のマイクロプロセッサー、およびあらゆる種類のデジタルコンピューターの任意の１若しくはそれ以上のプロセッサーが含まれる。一般に、プロセッサーは、読み取り専用メモリまたはランダムアクセスメモリ、あるいはその両方から命令とデータを受け取る。コンピューターの本質的な要素は、命令を実行するためのプロセッサーと、命令やデータを記憶するための１つ以上のメモリ装置である。一般に、コンピューターは、データを記憶するための１若しくはそれ以上の大容量記憶装置、例えば磁気ディスク、光磁気ディスク、光ディスクなどからもデータを受け取るか、その両方にデータを転送するように動作可能に結合される。コンピュータープログラム命令およびデータを具現化するのに適した情報担体には、一例として、半導体メモリ装置、例えば、電気的にプログラム可能な読み出し専用メモリまたはＲＯＭ（ＥＰＲＰＭ）、電気的に消去可能なプログラム可能ＲＯＭ（ＥＥＰＲＯＭ）、フラッシュメモリ装置、およびデータ記憶ディスク（例えば、磁気ディスク、内蔵ハードディスク、またはリムーバブルディスク、光磁気ディスク、およびＣＤ－ＲＯＭおよびＤＶＤ－ＲＯＭディスク）を含む、あらゆる形態の不揮発性メモリが含まれる。前記プロセッサーとメモリは、特殊用途の論理回路によって補足されるか、または特殊用途の論理回路に組み込まれてもよい。

【０１２９】

当業者であれば、情報および信号は、様々な異なる技術および技法のいずれかを用いて表すことができることを理解する。例えば、上記の説明を通して参照されるデータ、命令、コマンド、情報、信号、ビット、シンボル、およびチップは、電圧、電流、電磁波、磁場または粒子、光場または粒子、またはそれらの任意の組み合わせによって表される可能性がある。

【０１３０】

当業者は、本明細書に開示された実施形態に関連して説明された様々な例示的論理ブロック、モジュール、回路、およびアルゴリズムステップが、電子ハードウェア、コンピュータソフトウェア、またはその両方の組み合わせとして実装され得ることをさらに理解する。ハードウェアおよびソフトウェアのこの交換可能性を明確に説明するために、様々な例示的構成要素、ブロック、モジュール、回路、および工程を、それらの機能の観点から一般的に上述してきた。このような機能がハードウェアとして実装されるかソフトウェアとして実装されるかは、システム全体に課される特定の用途および設計上の制約に依存する。熟練した当業者であれば、特定の用途ごとに様々な方法で説明した機能を実装することができるが、そのような実装の決定は、本開示の範囲からの逸脱を引き起こすものとして解釈されるべきではない。ソフトウェアモジュールは、ランダムアクセスメモリ（ＲＡＭ）、フラッシュメモリ、ＲＯＭ、ＥＰＲＯＭ、ＥＥＰＲＯＭ、レジスタ、ハードディスク、リムーバブルディスク、ＣＤ－ＲＯＭ、または当技術分野で知られている他の形態の記憶媒体に存在することができる。例示的な記憶媒体は、プロセッサーが記憶媒体から情報を読み出し、記憶媒体に情報を書き込むことができるように、プロセッサーに結合される。別の方法として、記憶媒体はプロセッサーと一体であってもよい。すなわち、プロセッサーと記憶媒体は、集積回路内に存在してもよいし、ディスクリート構成要素として実装されてもよい。

【０１３１】

本明細書で使用される場合、「機械可読媒体」とは、命令およびデータを一時的または永続的に記憶することができる装置を意味し、ランダムアクセスメモリ（ＲＡＭ）、読み取り専用メモリ（ＲＯＭ）、バッファメモリ、フラッシュメモリ、光媒体、磁気媒体、キ

ャッシュメモリ、他のタイプの記憶装置（例えば、ＥＥＰＲＯＭ（Ｒｒａｓａｂｌｅ Ｐｒｏｇｒａｍｍａｂｌｅ Ｒｅａｄ－Ｏｎｌｙ Ｍｍｅｍｏｒｙ））、および／またはそれらの任意の適切な組み合わせを含むが、これらに限定されない。「機械可読媒体」という用語は、プロセッサー命令を記憶することができる単一の媒体または複数の媒体（例えば、集中型データベースまたは分散型データベース、または関連するキャッシュおよびサーバー）を含むものとみなされるべきである。また、「機械可読媒体」という用語は、１若しくはそれ以上のプロセッサーによる実行のための命令を記憶することができ、その命令が１若しくはそれ以上のプロセッサーによって実行されると、１若しくはそれ以上のプロセッサーに、本明細書に記載される方法論のいずれか１若しくはそれ以上を実行させるような、任意の媒体、または複数の媒体の組み合わせを含むものとみなされるものとする。したがって、「機械可読媒体」とは、単一の記憶装置または装置、および複数の記憶装置または装置を含む「クラウドベースの」記憶システムまたは記憶ネットワークを指す。本明細書で使用する「機械可読媒体」という用語は、信号そのものを除外する。

【０１３２】

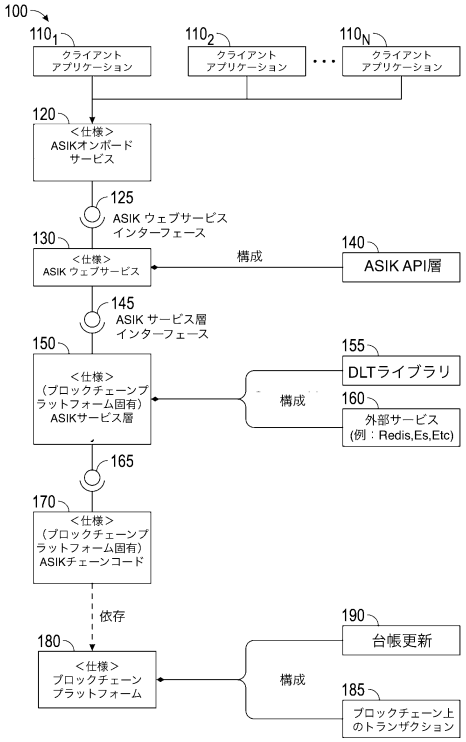
上述した説明および図は、例示のみを意図したものであり、添付の特許請求の範囲に記載されている場合を除き、例示的な実施形態をいかなる形でも限定することを意図したものではない。上述してきた様々な例示的実施形態の様々な要素の様々な技術的側面は、多数の他の方法で組み合わせることができ、その全てが本開示の範囲内であると考えられることに留意されたい。

【０１３３】

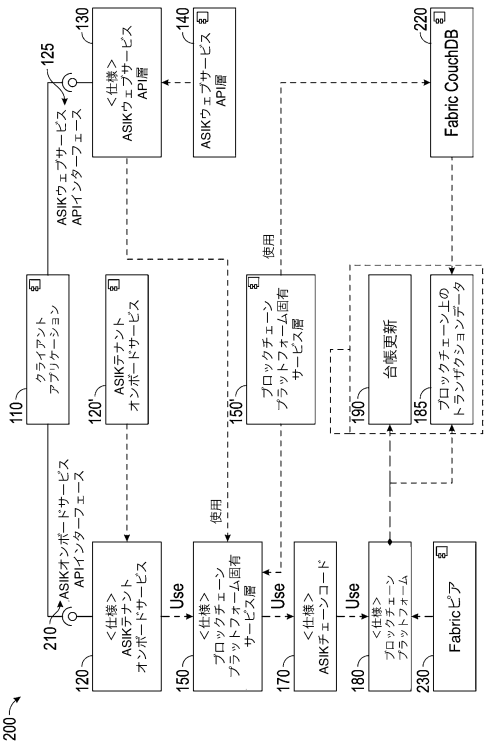
したがって、例示的な実施形態が例示目的で開示されているが、当業者であれば、様々な変更、追加、および置換が可能であることを理解するであろう。したがって、本開示は、上述の実施形態に限定されるものではなく、添付の特許請求の範囲の範囲内において、それらの均等物の全範囲と共に変更することができる。

【図面】

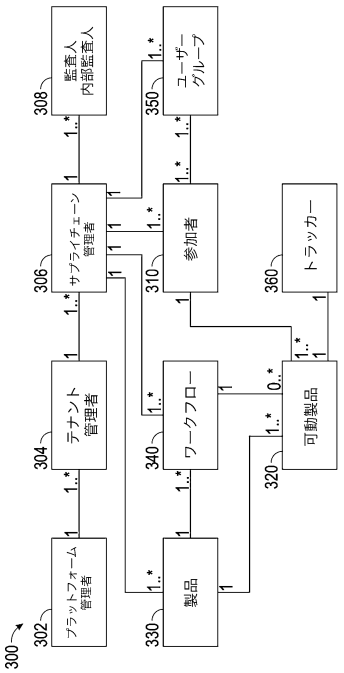
【図１】



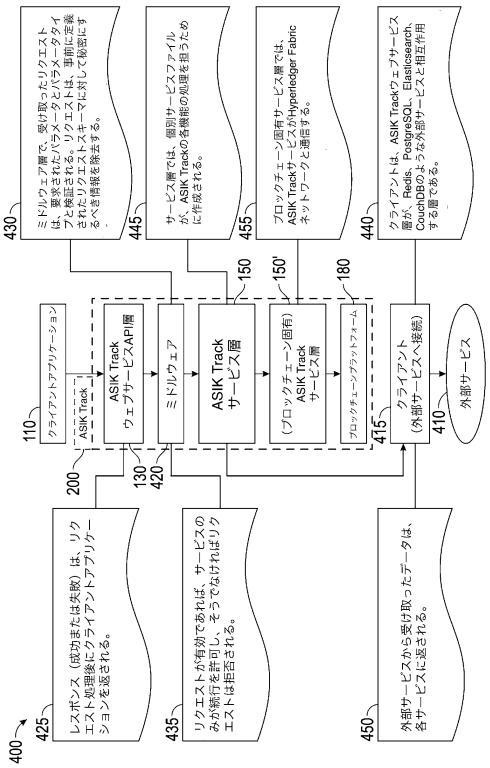

【図２】



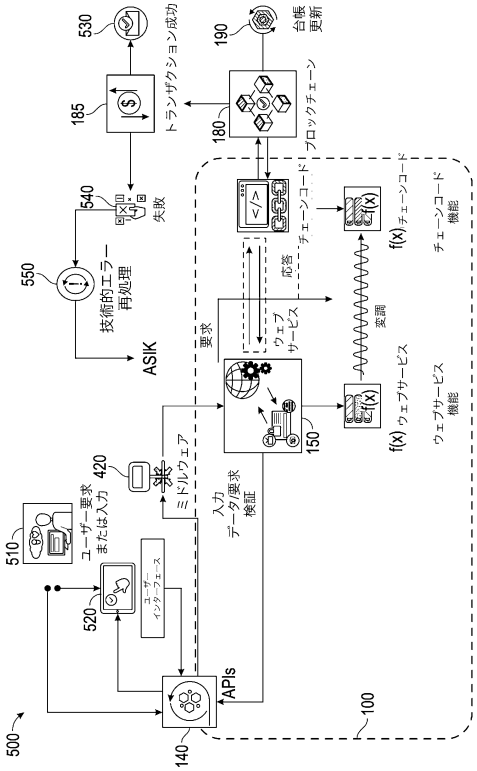
【図３】


300
302 プラットフォーム管理者
304 テナント管理者
306 サプライチェーン管理者
308 監査人 内部監査人
310 参加者
320 可動製品
330 製品
340 ワークフロー
350 ユーザーグループ
360 トラッカー
1
1..*

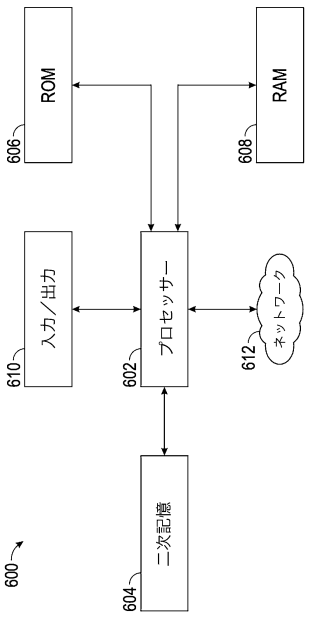
【図４】


400
410 外部サービス
415 クライアント（外部サービスへ接続）
420
425 レスポンス（成功または失敗）は、リクエスト処理後にクライアントアプリケーションを返される。
430 ミドルウェア層で、受け取ったリクエストは、要求されたパラメータとパラメータタイプと検証される。リクエストは、事前に定義されたリクエストスキーマに対して検証されるべき情報を除去する。
435 リクエストが有効であれば、サービスのみが続行を許可し、そうでなければリクエストは拒否される。
440 クライアントは、ASIK Trackウェブサービス層が、Redis、PostgreSQL、Elasticsearch、CouchDBのような外部サービスと相互作用する層である。
445 サービス層では、個別サービスファイルが、ASIK Trackの各機能の処理を担うために作成される。
450 外部サービスから受け取ったデータは、各サービスに返される。
455 ブロックチェーン固有サービス層では、ASIK TrackサービスがHyperledger Fabricネットワークと相互する。
110 クライアントアプリケーション
130 ASIK Track ウェブサービスAPI層
150 ASIK Track サービス層
150' (ブロックチェーン固有) ASIK Track サービス層
180 ブロックチェーンプラットフォーム
200 ASIK Track
ミドルウェア

【図５】


500
100
140 APIs
510 ユーザー要求または入力
520 ユーザーインターフェース
420 ASIK ミドルウェア
入力データ/要求検証
150
ウェブサービス
f(x) ウェブサービス
ウェブサービス機能
変調
応答
要求
f(x) チェーンコード
チェーンコード機能
180 ブロックチェーン
190 台帳更新
185 トランザクション成功
530
540 失敗
550 技術的エラー再処理
ASIK

【図６】


600
602 プロセッサー
604 二次記憶
606 ROM
608 RAM
610 入力／出力
612 ネットワーク

【国際調査報告】

**INTERNATIONAL SEARCH REPORT**

International application No.
**PCT/CA2023/050696**

A. CLASSIFICATION OF SUBJECT MATTER

IPC: ***G06F 9/44*** (2018.01), ***G06F 9/54*** (2006.01), ***G06F 16/27*** (2019.01)

CPC: **G06F 9/44** (2020.01), G06F 9/541 (2020.01), G06F 16/27 (2020.01)

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Keywords used across the whole IPC

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic database(s) consulted during the international search (name of database(s) and, where practicable, search terms used)
Questel orbit, Google Patent:

Keywords: blockchain, transaction, request, layer,dispatch

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y | US 2017/0140408 A1 (WUEHLER.) 18 May 2017 (18-05-2017)<br>See abstract, paragraph 77, 80, 81, 40, figure 9; | 1-25 |
| Y | US 7088995 B2 (RAO.) 8 August 2006 (08-08-2006)<br>See abstract, col.9, lines 47-64, col.2, lines 14-35, col.13, lines 13-39, col.15, lines 18-23 | 1-25 |

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

| | | | |
|---|---|---|---|
| * | Special categories of cited documents: | "T" | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| "A" | document defining the general state of the art which is not considered to be of particular relevance | "X" | document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| "D" | document cited by the applicant in the international application | "Y" | document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| "E" | earlier application or patent but published on or after the international filing date | "&" | document member of the same patent family |
| "L" | document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | | |
| "O" | document referring to an oral disclosure, use, exhibition or other means | | |
| "P" | document published prior to the international filing date but later than the priority date claimed | | |

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 29 June 2023 (29-06-2023) | 29 June 2023 (29-06-2023) |
| Name and mailing address of the ISA/CA<br>Canadian Intellectual Property Office<br>Place du Portage I, C114 - 1st Floor, Box PCT<br>50 Victoria Street<br>Gatineau, Quebec K1A 0C9<br>Facsimile No.: 819-953-2476 | Authorized officer<br>Li Pan (819) 639-8273 |

**INTERNATIONAL SEARCH REPORT**
Information on patent family members

International application No.
**PCT/CA2023/050696**

| Patent Document Cited in Search Report | Publication Date | Patent Family Member(s) | Publication Date |
|---|---|---|---|
| US2017140408A1 | 18 May 2017 (18-05-2017) | None | |
| US2003139174A1 | 24 July 2003 (24-07-2003) | US7088995B2 | 08 August 2006 (08-08-2006) |

フロントページの続き

,MC,ME,MK,MT,NL,NO,PL,PT,RO,RS,SE,SI,SK,SM,TR),OA(BF,BJ,CF,CG,CI,CM,GA,GN,GQ,GW,KM,ML,MR,NE,SN,TD,TG),AE,AG,AL,AM,AO,AT,AU,AZ,BA,BB,BG,BH,BN,BR,BW,BY,BZ,CA,CH,CL,CN,CO,CR,CU,CV,CZ,DE,DJ,DK,DM,DO,DZ,EC,EE,EG,ES,FI,GB,GD,GE,GH,GM,GT,HN,HR,HU,ID,IL,IN,IQ,IR,IS,IT,JM,JO,JP,KE,KG,KH,KN,KP,KR,KW,KZ,LA,LC,LK,LR,LS,LU,LY,MA,MD,MG,MK,MN,MU,MW,MX,MY,MZ,NA,NG,NI,NO,NZ,OM,PA,PE,PG,PH,PL,PT,QA,RO,RS,RU,RW,SA,SC,SC,SD,SE,SG,SK,SL,ST,SV,SY,TH,TJ,TM,TN,TR,TT,TZ,UA,UG,US,UZ,VC,VN,WS,ZA,ZM,ZW

（特許庁注：以下のものは登録商標）

１．ＪＡＶＡＳＣＲＩＰＴ

【要約の続き】

ーに戻る。

【選択図】　図４