

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7658685号
(P7658685)

(45)発行日 令和7年4月8日(2025.4.8)

(24)登録日 令和7年3月31日(2025.3.31)

(51)国際特許分類

G 0 6 F 8/60 (2018.01)

F I G 0 6 F 8/60

請求項の数 20 (全25頁)

(21)出願番号	特願2023-500389(P2023-500389)	(73)特許権者	390009531
(86)(22)出願日	令和3年6月24日(2021.6.24)		インターナショナル・ビジネス・マシー
(65)公表番号	特表2023-532370(P2023-532370		ンズ・コーポレーション
	A)		INTERNATIONAL BUSI
(43)公表日	令和5年7月27日(2023.7.27)		NESS MACHINES CORPO
(86)国際出願番号	PCT/IB2021/055603		RATION
(87)国際公開番号	WO2022/009011		アメリカ合衆国10504 ニューヨー
(87)国際公開日	令和4年1月13日(2022.1.13)		ク州 アーモンク ニュー オーチャード
審査請求日	令和5年11月14日(2023.11.14)		ロード
(31)優先権主張番号	16/921,381		New Orchard Road, A
(32)優先日	令和2年7月6日(2020.7.6)		rmonk, New York 105
(33)優先権主張国・地域又は機関	米国(US)	(74)代理人	04, United States of
			America
			100112690
			弁理士 太佐 種一
			最終頁に続く

(54)【発明の名称】 クラウドコンピューティング環境における非同期操作の管理

(57)【特許請求の範囲】

【請求項1】

サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I / O ベースのプログラムコードの実行のためのコンピュータ実装方法であって、前記方法は、前記プログラムコードの非同期操作を実行することであって、前記非同期操作は補完関数で完了される、実行することと、
前記非同期操作に関連する処理をハイバネーションすることと、
関連するシステムメモリを除く、前記処理のクラウドランタイムプラットフォーム関連リソースを解放することと、
前記処理のハイバネーション中に、前記非同期操作を実行することと、
前記非同期操作の完了後、前記処理に対して開始された補完関数を遮断することと、
前記補完関数に追加プログラムコードとデータの少なくとも1つを注入することと、
前記処理をハイバネーション解除し、前記処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、
前記非同期操作の結果データと、前記処理に対する前記追加プログラムコードと前記データの少なくとも1つとを返す前記補完関数を実行することであって、前記サーバレスクラウドランタイムプラットフォーム上での前記シングルスレッド、非同期 I / O ベースのプログラムコードの前記実行が有効化される、実行することと、
を含む方法。

【請求項2】

前記処理をハイバネーション解除することは、前記処理のリソースの配置ルーチンを実行することを含む、請求項 1 に記載の方法。

【請求項 3】

前記配置ルーチンは、前記クラウドランタイムプラットフォームのリソースの地理的可用性の関数である、請求項 2 に記載の方法。

【請求項 4】

前記配置ルーチンはさらに、前記非同期操作の位置に基づいてハイバネーション解除すべき前記処理のリソース配置を決定することを含む、請求項 2 または 3 に記載の方法。

【請求項 5】

前記ハイバネーションされた処理の位置に基づいて前記非同期操作によって要求されるリソースに対するリソース配置を決定することを含む、請求項 1 ～ 4 のいずれか 1 項に記載の方法。

【請求項 6】

前記非同期操作は、前記クラウドランタイムプラットフォームによってサービスとして実装される、請求項 1 ～ 5 のいずれか 1 項に記載の方法。

【請求項 7】

前記非同期操作は、長時間動作する非同期入力／出力操作、タイマ操作、または入力操作の待機操作である、請求項 1 ～ 6 のいずれか 1 項に記載の方法。

【請求項 8】

前記方法は、
前記非同期操作の前記実行を開始した配置ルーチンによって、前記非同期操作に関連付けられた前記データ分類とそれぞれのパラメータとに基づいて、ハイバネーション解除すべき前記処理のためのリソース配置を決定することを含む、請求項 1 ～ 7 のいずれか 1 項に記載の方法。

【請求項 9】

サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行のためのランタイムシステムであって、前記ランタイムシステムは、

メモリと、

前記メモリと通信するプロセッサとを含み、前記プロセッサは、
前記プログラムコードの非同期操作を実行することであって、前記非同期操作は補完関数で完了される、実行することと、

前記非同期操作に関連する処理をハイバネーションすることと、
関連するシステムメモリを除く、前記処理のクラウドランタイムプラットフォーム関連リソースを解放することと、

前記処理のハイバネーション中に、前記非同期操作を実行することと、
前記非同期操作の完了後、前記処理に対して開始された補完関数を遮断することと、
前記補完関数に追加プログラムコードとデータの少なくとも 1 つを注入することと、
前記処理をハイバネーション解除し、前記処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、

前記非同期操作の結果データと、前記処理に対する前記追加プログラムコードと前記データの少なくとも 1 つとを返す前記補完関数を実行することであって、前記シングルスレッド、非同期 I/O の前記実行が有効化される、実行することと、

を含む動作を実行するように構成される、

ランタイムシステム。

【請求項 10】

前記処理をハイバネーション解除することは、前記処理のリソースの配置ルーチンを実行することを含む、請求項 9 に記載のランタイムシステム。

【請求項 11】

10

20

30

40

50

前記配置ルーチンは、前記クラウドランタイムプラットフォームのリソースの地理的可用性の関数である、請求項 10 に記載のランタイムシステム。

【請求項 12】

前記配置ルーチンはさらに、前記非同期操作の位置に基づいてハイバネーション解除すべき前記処理のリソース配置を決定すること

を含む、請求項 10 または 11 に記載のランタイムシステム。

【請求項 13】

前記プロセッサはさらに、前記ハイバネーションされた処理の位置に基づいて前記非同期操作によって要求されるリソースに対するリソース配置を決定すること

を含む動作を実行するように構成される、請求項 9 ～ 12 のいずれか 1 項に記載のランタイムシステム。

10

【請求項 14】

前記非同期操作は、前記クラウドランタイムプラットフォームによってサービスとして実装される、請求項 9 ～ 13 のいずれか 1 項に記載のランタイムシステム。

【請求項 15】

前記非同期操作は、長時間動作する非同期入力／出力操作、タイマ操作、または入力操作の待機操作である、請求項 9 ～ 14 のいずれか 1 項に記載のランタイムシステム。

【請求項 16】

サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行のための コンピュータプログラム であって、前記 コンピュータプログラム は、1 または複数のコンピューティングシステムまたはコントローラに動作を実行させ、前記動作は、

20

前記プログラムコードの非同期操作を実行することであって、前記非同期操作は補完関数で完了される、実行することと、

前記非同期操作に関連する処理をハイバネーションすることと、

関連するシステムメモリを除く、前記処理のクラウドランタイムプラットフォーム関連リソースを解放することと、

前記処理のハイバネーション中に、前記非同期操作を実行することと、

前記非同期操作の完了後、前記処理に対して開始された補完関数を遮断することと、

前記補完関数に追加プログラムコードとデータの少なくとも 1 つを注入することと、

30

前記処理をハイバネーション解除し、前記処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、

前記非同期操作の結果データと、前記処理に対する前記追加プログラムコードと前記データの少なくとも 1 つとを返す前記補完関数を実行することであって、前記シングルスレッド、非同期 I/O の前記実行が有効化される、実行することと、

を含む、コンピュータプログラム。

【請求項 17】

前記処理をハイバネーション解除することは、前記処理のリソースの配置ルーチンを実行することを含む、請求項 16 に記載の コンピュータプログラム。

【請求項 18】

40

前記配置ルーチンは、前記クラウドランタイムプラットフォームのリソースの地理的可用性の関数である、請求項 17 に記載の コンピュータプログラム。

【請求項 19】

前記配置ルーチンはさらに、前記非同期操作の位置に基づいてハイバネーション解除すべき前記処理のリソース配置を決定すること

を含む、請求項 17 または 18 に記載の コンピュータプログラム。

【請求項 20】

前記コントローラは、さらに、前記ハイバネーションされた処理の位置に基づいて前記非同期操作によって要求されるリソースに対するリソース配置を決定すること

を含む動作を実行するように構成される、

50

請求項 16～19 のいずれか 1 項に記載のコンピュータプログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本開示は、一般に、クラウドコンピューティング環境における非同期操作の管理に関し、より具体的には、サーバレスクラウドランタイム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行に関する。

【背景技術】

【0002】

クラウドコンピューティングのシステム、環境およびフレームワークは、しばしばリソースや設備投資の面で柔軟性があるため、企業コンピューティングの主流となっている。さらに、サービス指向アーキテクチャ、仮想マシン、ステートレス運用、サーバなしでサービスを運用するオプション（サーバレスなど）の能力を探るために、多くのプログラミングモデルや運用モデルが開発され、しばしばオープンソースを用いて公開されている。

【0003】

マルチテナントのクラウド環境では、異なるワークロードが関連リソースを常に競合する可能性がある。クラウドのワークロードは、定義ごとにワークロードを実行するサービスがステートレスであるため、容易にスケールアウトすることが可能である。クラウドコンピューティング環境に投入されるワークロードに応じて、クラウドプロバイダはサービスインスタンスをスケールアウトしなければならない可能性があり、仮想マシン（「VM」）などのリソースをスケジュールしなければならない可能性がある。入ってくるワークロードの典型的な分散メカニズムは、現在のワークロードが最も少ないターゲットリソース（例えば、基礎となるハードウェアシステム）を選択することによってよい。

【0004】

サーバレス、クラウドベースコンピューティングは、クラウドプロバイダが物理的または仮想的なコンピューティングリソースを提供する実行モデルである。このようなシナリオにおけるリソースの配分は、シングルスレッドまたはマルチスレッドタイプであり得る選択されている実行ランタイムによっても決まる可能性がある。クライアントはサーバレスコンピューティング環境のリソースを申し込むおよび利用することができ、クラウドプロバイダは基盤となるインフラストラクチャの割り当てを管理することができる。

【0005】

特定の実行要件に有利な一方で、クラウドベースコンピューティングはいくつかの欠点も有し得る。1つの欠点は、異なるコーディング概念が要求され得ることであり、例えば、プログラムコードは、そのような環境で実行可能であるために、サーバレスフレームワークプリミティブの概念に適合させられなければならないかもしれないということである。一例としては、コードを単一の実行サイクルを表すスライスに分解し、これらのスライスを他の関数と一緒に連結することであってもよい。通常、サーバレス計算モデルを使用して既存のプログラムを変更せずに実行することはできない。

【0006】

米国特許出願公開第 2019/0339955 号 A1 には、第 1 のサービス上に位置するプログラムコード関数のインスタンスに対する関数チェックポイントを作成し、関数チェックポイントを使用して、第 2 のサーバ上にプログラムコード関数のインスタンスをロードするための技術が記載されている。

【0007】

さらに、米国特許出願公開第 2019/0384655 号 A1 には、サーバレスコンピューティング環境においてデータを処理するための方法および関連するシステムが記載されている。一実施形態では、複数のサーバレスコンピューティングプラットフォームに関連する機能に関連する情報および処理されるデータが、データソースから受信される。その後、データの処理に要求される処理操作が、データの特性およびデータに関連するドメインに基づいて決定される。処理操作を実行するためのターゲット機能が、信頼性ソース

10

20

30

40

50

の評価に基づいて特定され、ターゲット機能は、ターゲット機能に関連付けられたサーバレスコンピューティングプラットフォームにデータを転送することによって展開される。一実施形態において、要件は、データを処理するための最適なサーバレスコンピューティングプラットフォームを動的に選択することによって、サーバを作成および維持する必要性を排除し得る。

【0008】

しかし、シングルスレッド、非同期I/Oベースの言語で記述されたプログラムコードを、サーバレスコンピューティングのフレームワークで変更せずに利用できる技術が求められている。

【先行技術文献】

【特許文献】

【0009】

【文献】米国特許出願公開第2019/0339955号

【文献】米国特許出願公開第2019/0384655号

【発明の概要】

【0010】

本開示のいくつかの態様によれば、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行のためのコンピュータ実装方法が提供され得る。本方法は、プログラムコードの非同期操作を実行することを含んでよく、非同期操作は補完関数で完了される。本方法は、非同期操作に関連する処理をハイバネーションすることと、関連するシステムメモリを除く、処理のクラウドランタイムプラットフォーム関連リソースを解放することを含んでもよい。本方法は、処理のハイバネーション中に、非同期操作を実行することと、非同期操作の完了後、処理に対して開始された補完関数を遮断することと、補完関数に追加プログラムコードとデータの少なくとも1つを注入することとを含むこともできる。

【0011】

さらに、本方法は、処理をハイバネーション解除し、処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、非同期操作の結果データと、処理に対する追加プログラムコードとデータの少なくとも1つを返す補完関数を実行することとであって、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行が有効化される、実行することと、を含んでよい。

【0012】

本開示の別の態様によれば、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行のためのランタイムシステムが提供され得る。本システムは、メモリと、メモリと通信するプロセッサとを含み得る。プロセッサは、プログラムコードの非同期操作を実行することとであって、非同期操作は補完関数で完了される、実行することと、非同期操作に関連する処理をハイバネーションすることと、関連するシステムメモリを除く、処理のクラウドランタイムプラットフォーム関連リソースを解放することと、を含む操作を実行するように構成されてよい。

【0013】

いくつかの実施形態において、プロセッサは、処理のハイバネーション中に、非同期操作を実行することと、非同期操作の完了後、処理に対して開始された補完関数を遮断することと、補完関数に追加プログラムコードとデータの少なくとも1つを注入することと、を含む操作を実行するように構成されてよい。

【0014】

いくつかの実施形態において、プロセッサは、処理をハイバネーション解除し、処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、を含む操作を実行するように構成されてよい。いくつかの実施形態において、プロセッサは、非同期操作の結果データと、処理に対する追加プログラムコードとデータの少なくとも

10

20

30

40

50

1つとを返す補完関数を実行することであって、シングルスレッド、非同期 I / O の実行が有効化される、実行することと、を含む操作を実行するように構成されてよい。

【0015】

サーバレスクラウドランタイムプラットフォーム上でシングルスレッド、非同期 I / O ベースのプログラムコードを実行するためのコンピュータ実装方法を提案し、複数の利点、貢献、および技術的效果を提供することができる。

【0016】

一例として、任意のタイプのシングルスレッドプログラムコード、例えば Node.js プログラムは、実行スライスをサーバレス呼び出しに透過的にマッピングすることによって、自動的にサーバレスコンピューティングモデルにマッピングされてもよい。したがって、本開示は、修正されていない JavaScript プログラムが、当初設計されていなかったサーバレス環境において実行されることを可能にし得る。本開示は、サーバレスコンピューティングの利点を、より伝統的なシングルスレッドで多かれ少なかれ線形に設計されたプログラムコードにももたらすものである。

【0017】

サーバレスコンピューティング環境のためにシングルスレッドのプログラムコードを書き直し、再設計する必要がなくなったため、ユーザの視点からはプログラムコードを無限に実行できるようになるが、サーバレスコンピューティングの視点からは依然として純粋なサーバレスアプリケーションである。これにより、2つの利点が得られる。1つ目は Node.js のようなシングルスレッドのプログラミング言語、2つ目はサーバレスコンピューティングが要求するリソースがより少なく済むという利点である。

【0018】

本開示の方法は、元のシングルスレッドプログラムを再コンパイルまたは再構築する必要なく、元のプログラムコードに任意のデータおよびプログラムロジックを注入することも可能にし得る。注入されたデータの例としては、現在のデータヒープに署名する暗号署名、およびプログラムコードの性能分析を支援し、ひいては特定のサーバレスランタイムおよびクラスタへの結果としてのロードプロファイルに依存する配置を最適化するプロファイリングデータが挙げられ得る（ただし、これらに限定されるものでない）。注入されたプログラムコードの例には、監査ロギングサービスにプログラムの実行状態を送信することも含まれる（ただし、これに限定されない）。

【0019】

本開示の方法の別の利点は、計算資源の利用可能性に依存して、中断されたシングルスレッドプログラムを別のクラウドクラスターまたは領域で再開する能力である。

【0020】

いくつかの実施形態では、シングルスレッド、非同期 I / O ベースのプログラムコードは、Node.js、JavaScript、TypeScript および Golang または同等のプログラミング言語から選択されるプログラミング言語で記述されてもよい。したがって、本開示は、シングルスレッド、同期 I / O ベースのプログラミングスタイルをサポートするほぼすべてのプログラミング言語とともに使用されてもよい。

【0021】

いくつかの実施形態では、ハイバネーション解除または再開することは、処理のリソースの配置ルーチンを実行することを含んでもよい。したがって、非同期操作を開始した開始プログラム処理によって要求されるリソースは、再確立されてもよく、それによって、サーバレスコンピューティングの概念がサポートされる。

【0022】

いくつかの実施形態では、配置ルーチンは、クラウドランタイムプラットフォームのリソースの地理的可用性の関数であってもよい。したがって、配置ルーチンのための有用な構成パラメータを選択することによって、関連サービスが実行される地理的な領域に影響を与えることができる。これにより、呼び出し元サービスと呼び出し先サービスとの間の距離が離れていることに起因するネットワーク遅延を低減することが可能となる。

【0023】

いくつかの実施形態では、配置ルーチンは、非同期操作の位置に基づいてハイバネーション解除すべき処理のリソース配置を決定することも含んでよい。位置は、地理的座標に基づいて決定されてもよく、例えば、関連サービス間の長距離ネットワーク接続を回避するためにリソース配置は「近く」に選択されてもよい。

【0024】

いくつかの実施形態では、方法は、ハイバネーションされた処理の位置に基づいて非同期操作によって要求されるリソースに対するリソース配置を決定することを含んでもよい。地理的座標も主要な役割を果たすことがある。これは、両方のサービス（特に、呼び出し元と呼び出し先）が同じクラウドコンピューティングセンターで実行され得ることを保証し得る。

10

【0025】

いくつかの実施形態では、サーバレスクラウドランタイムプラットフォームは、以下の技術のうちの1つに基づくことができる：Docker、Containerd、Linux（登録商標）Containers、Podman、Kubernetes、Knative、AWS Lambda、Google Cloud Functions、Azure Functions、およびAlibaba Cloud Function Compute。さらに、他の類似のサーバレスクラウドランタイムプラットフォームが使用されることもある。したがって、提案された概念は、基本的にあらゆるサーバレスクラウドランタイムフレームワークをサポートすることができる。

【0026】

20

いくつかの実施形態では、非同期操作は、長時間実行される非同期入力／出力（I／O）操作（例えば、ファイルシステム、データベース、または同様のものへの呼び出し）タイム操作、または入力操作の待機操作であってもよい。ただし、サーバレスコンピューティングのフレームワークにおけるプロミスに直接適用するのではなく、入力／出力を構成するものに適用する場合があることに留意されたい。非同期待ち操作についても同様に適用することができ、これらは一般的にプロミスにマッピングされる。

【0027】

いくつかの実施形態では、方法は、非同期操作の実行を開始（例えば、トリガ）した配置ルーチンによって、非同期操作に関連付けられたデータ分類とそれぞれのパラメータとに基づいて、ハイバネーション解除すべき処理のリソース配置を決定することも含んでもよい。このように、コードそのものや潜在的なメタデータだけでなく、非同期操作に関連するデータのタイプを利用して、リソース配置の機能を制御することができる。これにより、シングルスレッド、非同期I／Oベースのプログラムコードのコンテキストで使用される場合、ステートレスコンピューティングフレームワークのきめ細かな管理が可能となる場合がある。

30

【0028】

さらに、実施形態は、コンピュータまたは任意の命令実行システムによって、またはそれに関連して、使用するためのプログラムコードを提供するコンピュータ使用可能媒体またはコンピュータ読取可能媒体からアクセス可能な、関連するコンピュータプログラム製品の形態をとることができる。この説明の目的のために、コンピュータ使用可能媒体またはコンピュータ読取可能媒体は、命令実行システム、装置、またはデバイスによって、または関連して、使用するためのプログラムを格納、通信、伝播、または輸送する手段を含む可能性がある任意の装置であってよい。

40

【0029】

上記の概要は、本開示の図示された各実施形態またはすべての実施態様を説明することを意図するものではない。

【0030】

本開示に含まれる図面は、本明細書に組み込まれ、本明細書の一部を構成する。それらは、本開示の実施形態を例示し、説明とともに、本開示の原理を説明するのに役立つ。図面は、特定の実施形態を例示するものに過ぎず、本開示を限定するものではない。

50

【図面の簡単な説明】**【0031】**

【図1】本開示の態様による、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行のための、コンピュータ実装方法のブロック図である。

【図2】本開示の態様による、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行のためのコンピュータ実装方法のブロック図コンポーネントである。

【図3】本開示の態様による、サーバレスクラウドランタイムプラットフォーム上でシングルスレッド、非同期I/Oベースのプログラムコードを実行するためのランタイムシステムのブロック図である。

10

【図4A】本開示の態様による、クラウドコンピューティング環境を示す図である。

【図4B】本開示の態様による、抽象化モデルレイヤを示す図である。

【図5】本開示の態様による、本明細書に記載される1または複数の方法、ツール、およびモジュール、ならびに任意の関連機能を実施する際に使用され得る例示的なコンピュータシステムのハイレベルブロック図である。

【発明を実施するための形態】**【0032】**

本明細書に記載された実施形態は、様々な変更および代替形態に従うことができるが、その具体的な内容は、図面において例として示されており、詳細に説明されるであろう。しかしながら、説明された特定の実施形態は、限定的な意味でとられるものではないことが理解されるべきである。それどころか、その意図は、本開示の精神および範囲内に入る全ての修正物、等価物、および代替物を網羅することである。

20

【0033】

本開示の態様は、一般に、クラウドコンピューティング環境における非同期操作の管理に関し、より具体的には、サーバレスクラウドランタイム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行に関連するものである。本開示は、必ずしもそのような用途に限定されないが、本開示の様々な態様は、この文脈を用いた様々な例の議論を通じて理解され得る。

【0034】

本説明では、以下の慣習、用語もしくは表現またはその組み合わせを使用することがある。

30

【0035】

「シングルスレッド、非同期I/Oベースのプログラムコード」という用語は、部分的な機能のステートフルな実行を可能にする異なるプログラミングコンセプトの下で書かれたプログラムコードを指すために使用されてもよい。すなわち、変数のステータスは、関連するシングルスレッド、非同期I/Oベースのプログラムコードの基礎となる完全な処理に対して維持されるかもしれない。これは、ハイパーバイザー上で完全な仮想マシンを使用するプログラミングモデルに典型的なものであるかもしれない。

【0036】

「サーバレスクラウドランタイムプラットフォーム」という用語は、クラウドプロバイダがサーバを実行し、割り当てマシンリソースを動的に管理するクラウドコンピューティング実行モデルを指すために使用される場合がある。このクラウドコンピューティング実行モデルは、完全な仮想マシンを維持する必要がない場合があり、サーバレスコードは、時に、マイクロサービスを使用して展開されたコードと組み合わせて使用することができる。しかし、典型的には、アプリケーションのプログラムコードは、純粋にサーバレスパラダイムに合わせて書かれる場合があり、完全なプロビジョニングサーバに全く依存しない場合もある。

40

【0037】

「非同期操作」という用語は、特に非同期I/O、つまり送信またはトランザクション

50

が終了する前に他の処理の継続を許可する可能性のある入力／出力処理の形式を指すために使用される場合がある。この入力／出力処理の形式は、インテリジェントなオペレーティングシステムが制御するプログラム関連活動の部分的な並列化を可能にする場合がある。非同期操作は、ディスクや類似のデバイスなどの一般的に遅い I／O デバイスの時間を、I／O データに依存しないプログラムコードに使用するのに役立つ場合がある。

【0038】

「プログラムコード」という用語は、コンピュータシステム、仮想コンピュータシステム、またはサーバレスコンピューティング環境において実行されるデジタルコードを指すために使用されてもよい。プログラムコードは、人間が理解可能な言語で記述されたプログラムの結果であってもよい。プログラムは、プログラムコードとして実行されるために、コンパイル、解釈または中間コードおよびオーダーへの変換が行われてもよい。

10

【0039】

「補完関数」という用語は、I／O サービスや I／O コールなど、呼び出された非同期操作に関連するプログラムコードの末尾で開始されるシステムコールを指す場合もある。

【0040】

「処理のハイバネーション」（ディスクへのサスペンションとも呼ばれることがある）という用語は、コンピュータまたはコンピュータの関連する部分サブシステムの状態を保持したまま電源を切ることを指すために使用されることがある。ハイバネーションに際して、コンピュータまたは関連するリソースは、その状態を不揮発性ストレージに保存することができる。受信時またはハイバネーション解除時には、コンピュータはハイバネーションに入る前に動作していた状態となる。また、古典的なハイバネーション処理とは対照的に、本明細書に開示される概念では、ランダムアクセスメモリ（例えば、実際の処理に関連するシステムメモリ）のコンテンツがディスクにサスペンドされない場合があることに留意されたい。

20

【0041】

「追加プログラムコードとデータの少なくとも1つを注入する」という用語は、非同期操作の結果が呼び出しルーチンに返されるだけでなく、追加のデータ、特に実データ形式のデータまたはプログラムコード形式のデータを呼び出しルーチンに返してもよいことを意味する場合がある。

【0042】

「ハイバネーション解除」とは、ハイバネーションする前と同じ状態情報でコンピュータの操作を再開させることを指す場合がある。

30

【0043】

「Node.js」という用語は、Web ブラウザの外で JavaScript コードを実行する既知のオープンソース、処理、プラットフォームの JavaScript ランタイム環境を指すために使用される場合がある。Node.js は、開発者が JavaScript を使ってコマンドラインツールを書くこと、および、サーバーサイドスクリプティング（例えば、ページがユーザの Web ブラウザに送られる前に動的な Web ページのコンテンツを生成するためにスクリプトをサーバーサイドで実行すること）のために使用することを可能にする。その結果、Node.js は、サーバーサイドとクライアントサイドのスクリプトのために異なる言語ではなく、単一のプログラミング言語を中心にウェブアプリケーション開発を統一する「どこでも JavaScript」パラダイムを示すかもしれない。

40

【0044】

「JavaScript（一般に“JS”と略される）」という用語は、ECMAScript 仕様に準拠したプログラミング言語を指す場合に使用されることがある。これは高水準の、しばしばジャストインタimeにコンパイルされるプログラミング言語である。

【0045】

「TypeScript」という用語は、Microsoft 社によって開発・保守されているオープンソースのプログラミング言語を指す場合に使用されることがある。この用語は JavaScript のスーパーセットを表し、プログラミング言語にオプションの静的型付けを追加し

50

ている。

【0046】

「Golang（または単に「Go」）」という用語は、Googleが設計・保守する既知の静的型付けコンパイル型プログラミング言語を指す場合に使用されることがある。このプログラミング言語はC言語に似ているが、メモリ安全性、ガベージコレクション、構造的型付け、CSPスタイルの並行処理を備えている。

【0047】

「配置ルーチン」という用語は、設定パラメータを伴う可能性のある非同期操作を開始するシステムコールを指す場合に使用されることがある。

【0048】

「Docker」、「Containerd」、「LXC」、「Podman Kubernetes」、「Knative」、「AWS Lambda」、「Google Cloud Functions」、「Azure Functions」、「Alibaba Cloud Function Compute」という用語は、サーバレスコンピューティングのためのフレームワークを指す場合に使用されることがある。これらのサーバレスコンピューティングのフレームワークのバージョンは、提案された概念の文脈で使用されることがある。上述したサーバレスコンピューティングのフレームワークの名称の多くは、「コンテナ化された（containerized）」サービスの実行に基づくものである。

【0049】

「クラウドコンピューティング」という用語、および「クラウドサービス環境」、「クラウドコンピューティング環境」、「クラウドコンピューティングフレームワーク」という用語は、この文脈では、設定可能なコンピューティングリソースの共有プール（例えばネットワーク、ネットワーク帯域幅、サーバ、処理、メモリ、記憶装置、アプリケーション、仮想マシンおよびサービス）へ、簡便かつオンデマンドのネットワークアクセスを可能にするためのサービス提供のモデルであり、リソースは、最小限の管理労力または最小限のサービスプロバイダとのやり取りによって速やかに準備（provision）およびリリースできるものとして解釈される場合がある。このクラウドモデルは、少なくとも5つの特性、少なくとも3つのサービスモデル、および少なくとも4つの実装モデルを含むことがある。

【0050】

特性は以下の通りである。

【0051】

オンデマンド・セルフサービス：クラウドの消費者は、サービスプロバイダとの人的な対話を必要することなく、必要に応じて自動的に、サーバ時間やネットワークストレージなどのコンピューティング能力を一方的に準備することができる。

【0052】

ブロード・ネットワークアクセス：コンピューティング能力はネットワーク経由で利用可能であり、また、標準的なメカニズムを介してアクセスできる。それにより、異種のシンまたはシッククライアントプラットフォーム（例えば、携帯電話、ラップトップ、PDA）による利用が促進される。

【0053】

リソースプーリング：プロバイダのコンピューティングリソースはプールされ、マルチテナントモデルを利用して複数の消費者に提供される。様々な物理リソースおよび仮想リソースが、需要に応じて動的に割り当ておよび再割り当てされる。一般に消費者は、提供されたリソースの正確な部分を管理または把握していないため、部分非依存（portion independence）の感覚がある。ただし消費者は、より高い抽象レベル（例えば、国、州、データセンタ）では部分を特定可能な場合がある。

【0054】

迅速な柔軟性（elasticity）：コンピューティング能力は、迅速かつ柔軟に準備することができるため、場合によっては自動的に、直ちにスケールアウトし、また、速やかにリリースされて直ちにスケールインすることができる。消費者にとって、準備に利用可能な

10

20

30

40

50

コンピューティング能力は無制限に見える場合が多く、任意の時間に任意の数量で購入することができる。

【0055】

測定されるサービス：クラウドシステムは、サービスの種類（例えば、ストレージ、処理、帯域幅、アクティブユーザアカウント）に適したある程度の抽象化レベルでの測定機能を活用して、リソースの使用を自動的に制御し最適化する。リソース使用量を監視、制御、および報告して、利用されるサービスのプロバイダおよび消費者の両方に透明性を提供することができる。

【0056】

サービスモデルは以下の通りである。

【0057】

サービスとしてのソフトウェア（SaaS）：消費者に提供される機能は、クラウドインフラストラクチャ上で動作するプロバイダのアプリケーションを利用することである。当該そのアプリケーションは、ウェブブラウザ（例えばウェブメール）などのシンクライアントインタフェースを介して、各種のクライアント装置からアクセスできる。消費者は、ネットワーク、サーバ、オペレーティングシステム、ストレージや、個別のアプリケーション機能さえも含めて、基礎となるクラウドインフラストラクチャの管理や制御は行わない。ただし、ユーザ固有の限られたアプリケーション構成の設定はその限りではない。

【0058】

サービスとしてのプラットフォーム（PaaS）：消費者に提供される機能は、プロバイダによってサポートされるプログラム言語およびツールを用いて、消費者が作成または取得したアプリケーションを、クラウドインフラストラクチャに展開（deploy）することである。消費者は、ネットワーク、サーバ、オペレーティングシステム、ストレージを含む、基礎となるクラウドインフラストラクチャの管理や制御は行わないが、展開されたアプリケーションを制御でき、かつ場合によってはそのホスティング環境の構成も制御できる。

【0059】

サービスとしてのインフラストラクチャ（IaaS）：消費者に提供される機能は、オペレーティングシステムやアプリケーションを含む任意のソフトウェアを消費者が展開および実行可能な、プロセッサ、ストレージ、ネットワーク、および他の基本的なコンピューティングリソースを準備することである。消費者は、基礎となるクラウドインフラストラクチャの管理や制御は行わないが、オペレーティングシステム、ストレージ、および展開されたアプリケーションを制御でき、かつ場合によっては一部のネットワークコンポーネント（例えばホストファイアウォール）を部分的に制御できる。

【0060】

展開モデルは以下の通りである。

【0061】

プライベートクラウド：このクラウドインフラストラクチャは、特定の組織専用で運用される。このクラウドインフラストラクチャは、当該組織または第三者によって管理することができ、オンプレミスまたはオフプレミスで存在することができる。

【0062】

コミュニティクラウド：このクラウドインフラストラクチャは、複数の組織によって共有され、共通の関心事（例えば、ミッション、セキュリティ要件、ポリシー、およびコンプライアンス）を持つ特定のコミュニティをサポートする。このクラウドインフラストラクチャは、当該組織または第三者によって管理することができ、オンプレミスまたはオフプレミスで存在することができる。

【0063】

パブリッククラウド：このクラウドインフラストラクチャは、不特定多数の人々や大規模な業界団体に提供され、クラウドサービスを販売する組織によって所有される。

【0064】

10

20

30

40

50

ハイブリッドクラウド：このクラウドインフラストラクチャは、2つ以上のクラウドモデル（プライベート、コミュニティまたはパブリック）を組み合わせたものとなる。それぞれのモデル固有の実体は保持するが、標準または個別の技術によってバインドされ、データとアプリケーションの可搬性（例えば、クラウド間の負荷分散のためのクラウドバースティング）を実現する。

【0065】

クラウドソフトウェアは、ステートレス性（statelessness）（例外あり）、低結合性（low coupling）、モジュール性（modularity）および意味論的相互運用性（semantic interoperability）に重点を置いたサービス指向で、クラウドパラダイムを最大限に活用することに留意されたい。

【0066】

本開示のいくつかの態様によれば、サーバレスクラウドランタイムプラットフォーム上でシングルスレッド、非同期 I/O ベースのプログラムコードを実行するためのコンピュータ実装方法が提供され得る。本方法は、プログラムコードの非同期操作を実行することを含んでよく、非同期操作は、補完関数で完了される。本方法は、非同期操作に関連する処理をハイバネーションすることと、関連するシステムメモリを除く、処理のクラウドランタイムプラットフォーム関連リソースを解放することを含んでもよい。この方法は、処理のハイバネーション中に、非同期操作を実行することと、非同期操作の完了後、処理に対して開始された補完関数を遮断することと、補完関数に追加プログラムコードとデータの少なくとも1つを注入することを含むこともできる。

【0067】

さらに、本方法は、処理をハイバネーション解除し、処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、非同期操作の結果データと、処理に対する追加プログラムコードとデータの少なくとも1つとを返す補完関数を実行することを含んでよく、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行が有効化されてもよい。

【0068】

本開示の別の態様によれば、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行のためのランタイムシステムが提供され得る。本システムは、メモリと、メモリと通信するプロセッサとを含んでもよい。プロセッサは、プログラムコードの非同期操作を実行することであって、非同期操作は補完関数で完了される、実行することと、非同期操作に関連する処理をハイバネーションすることと、関連システムメモリを除く処理のクラウドランタイムプラットフォーム関連リソースを解放することを含む操作を行うように構成されていてもよい。

【0069】

いくつかの実施形態において、プロセッサは、処理のハイバネーション中に、非同期操作を実行することと、非同期操作の完了後、処理に対して開始された補完関数を遮断することと、補完関数に追加プログラムコードとデータの少なくとも1つを注入することを含む操作を実行するように構成されてもよい。

【0070】

いくつかの実施形態において、プロセッサは、処理をハイバネーション解除し、処理の解放されたクラウドランタイムプラットフォーム関連リソースを再割り当てすることと、を含む操作を実行するように構成されてもよい。いくつかの実施形態において、プロセッサは、非同期操作の結果データと、処理に対する追加プログラムコードとデータの少なくとも1つとを返す補完関数を実行することを含む操作を実行するよう構成されてよく、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行が有効化されてもよい。

【0071】

サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期 I/O ベースのプログラムコードの実行のためのコンピュータ実装方法を提案し、複数の利点

10

20

30

40

50

、貢献、および技術的效果を提供することができる。

【0072】

一例として、任意のタイプのシングルスレッドプログラムコード、例えば、Node.jsプログラムは、実行スライスサーバレス呼び出しに透過的にマッピングすることによって、自動的にサーバレスコンピューティングモデルにマッピングされてもよい。したがって、本開示は、修正されていないJavaScriptプログラムが、当初設計されていなかったサーバレス環境で実行されることを可能にし得る。本開示は、サーバレスコンピューティングの利点を、より伝統的なシングルスレッドで多かれ少なかれ線形に設計されたプログラムコードにももたらすものである。

【0073】

サーバレスコンピューティング環境のためにシングルスレッドのプログラムコードを書き直し、再設計する必要がなくなったため、ユーザの視点からはプログラムコードを無限に実行できるようになるが、サーバレスコンピューティングの視点からは依然として純粋なサーバレスアプリケーションである。これにより、2つの利点を得られる。1つ目はNode.jsのようなシングルスレッドのプログラミング言語、2つ目はサーバレスコンピューティングが要求するリソースがより少なく済むという利点である。

【0074】

本開示の方法は、元のシングルスレッドプログラムを再コンパイルまたは再構築する必要なく、元のプログラムコードに任意のデータおよびプログラムロジックを注入することも可能にし得る。注入されたデータの例としては、現在のデータヒープに署名する暗号署名、およびプログラムコードの性能分析を支援し、ひいては特定のサーバレスランタイムおよびクラスタへの結果としてのロードプロファイルに依存する配置を最適化するプロファイリングデータが挙げられ得る（ただし、これらに限定されるものでない）。注入されたプログラムコードの例には、監査ロギングサービスにプログラムの実行状態を送信することも含まれる（ただし、これに限定されない）。

【0075】

本開示の方法の別の利点は、計算資源の利用可能性に依存して、中断されたシングルスレッド・プログラムを別のクラウドクラスタまたは領域で再開する能力である。

【0076】

いくつかの実施形態では、シングルスレッド、非同期I/Oベースのプログラムコードは、Node.js、JavaScript、TypeScriptおよびGolangまたは同等のプログラミング言語から選択されるプログラミング言語で記述されてもよい。したがって、本開示は、シングルスレッド、同期I/Oベースのプログラミングスタイルをサポートするほぼすべてのプログラミング言語とともに使用されてもよい。

【0077】

いくつかの実施形態では、ハイバネーション解除または再開は、処理のリソースの配置ルーチンを実行することを含んでもよい。したがって、非同期操作を開始した開始プログラムプロセスによって要求されるリソースは、再確立されてもよく、それによって、サーバレスコンピューティングの概念がサポートされる。

【0078】

いくつかの実施形態では、配置ルーチンは、クラウドランタイムプラットフォームのリソースの地理的可用性の関数であってもよい。したがって、配置ルーチンのための有用な構成パラメータを選択することによって、関連サービスが実行される地理的な領域に影響を与えることができる。これにより、呼び出し元サービスと呼び出し先サービスとの間の距離が離れていることに起因するネットワーク遅延を低減することが可能となる。

【0079】

いくつかの実施形態では、配置ルーチンは、非同期操作の位置に基づいてハイバネーション解除すべき処理のリソース配置を決定することを含んでもよい。位置は、地理的座標に基づいて決定されてもよく、例えば、関連サービス間の長距離ネットワーク接続を回避するためにリソース配置は「近く」に選択されてもよい。

10

20

30

40

50

【0080】

いくつかの実施形態では、方法は、ハイバネーションされた処理の位置に基づいて非同期操作によって要求されるリソースに対するリソース配置を決定することを含んでもよい。地理的座標も主要な役割を果たすことがある。これは、両方のサービス（特に、呼び出し元と呼び出し先）が同じクラウドコンピューティングセンターで実行され得ることを保証し得る。

【0081】

いくつかの実施形態では、サーバレスクラウドランタイムプラットフォームは、以下の技術のうちの1つに基づくことができる：Docker、Containerd、Linux Containers、Podman Kubernetes、Knative、AWS Lambda、Google Cloud Functions、Azure Functions、およびAlibaba Cloud Function Compute。さらに、他の類似のサーバレスクラウドランタイムプラットフォームが使用されることもある。したがって、提案された概念は、基本的にあらゆるサーバレスクラウドランタイムフレームワークをサポートすることができる。

【0082】

いくつかの実施形態では、非同期操作は、長時間動作する非同期入力／出力（I／O）操作（例えば、ファイルシステム、データベース、または同様のものへの呼び出し）タイマ操作、または入力操作の待機操作であってもよい。ただし、サーバレスコンピューティングのフレームワークにおけるプロミスに直接適用するのではなく、入力／出力を構成するものに適用する場合があることに留意されたい。非同期待ち操作についても同様に適用することができ、これらは一般的にプロミスにマッピングされる。

【0083】

いくつかの実施形態では、方法は、非同期操作の実行を開始（例えば、トリガ）した配置ルーチンによって、非同期操作に関連付けられたデータ分類とそれぞれのパラメータとに基づいて、ハイバネーション解除すべき処理のリソース配置を決定することも含んでいてもよい。このように、コードそのものや潜在的なメタデータだけでなく、非同期操作に関連するデータのタイプを利用して、リソース配置の機能を制御することができる。これにより、シングルスレッド、非同期I／Oベースのプログラムコードのコンテキストで使用される場合、ステートレスコンピューティングフレームワークのきめ細かな管理が可能となる場合がある。

【0084】

さらに、実施形態は、コンピュータまたは任意の命令実行システムによって、またはそれに関連して、使用するためのプログラムコードを提供するコンピュータ使用可能媒体またはコンピュータ読取可能媒体からアクセス可能な、関連するコンピュータプログラム製品の形態をとることができる。この説明の目的のために、コンピュータ使用可能媒体またはコンピュータ読取可能媒体は、命令実行システム、装置、またはデバイスによって、またはそれに関連して、使用するためのプログラムを格納、通信、伝播、または輸送する手段を含む可能性がある任意の装置であってもよい。

【0085】

以下、図について詳しく説明する。図中の指示は全て模式的なものである。最初に、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I／Oベースのプログラムコードの実行のためのコンピュータ実装方法の実施形態のブロック図が与えられる。その後、さらなる実施形態、ならびにサーバレスクラウド実行時プラットフォーム上でのシングルスレッド、非同期I／Oベースのプログラムコードの実行のためのランタイムシステムの実施形態が説明される。

【0086】

図1を参照すると、サーバレスクラウドコンピューティングランタイムプラットフォーム（例えば、Docker）上でのシングルスレッド、非同期I／Oベースのプログラムコード（例えば、Node.JS、JavaScript、TypeScriptまたはGolang）の実行のためのコンピュータ実装方法100の一実施形態のブロック図が示されている。プログラムコードは、コ

10

20

30

40

50

ンパイルされていてもよく、解釈可能なコードであっても、中間コードに基づくプログラムコードであってもよい。

【0087】

いくつかの実施形態では、方法100は、操作102で始まる。操作102において、プログラムコードの非同期操作が実行される。それにより、非同期操作は、補完関数で完了されることになる。代替的に、コールバック関数でも完了されることができる。

【0088】

いくつかの実施形態において、方法100は、操作104に進む。操作104では、非同期操作に関連する処理がハイバネーションされる。いくつかの実施形態では、方法100は、操作106に進み、関連するシステムメモリ（例えば、RAM）を除く、処理のクラウドランタイムプラットフォーム関連リソースが、解放（freed-up）または解放（released）される。したがって、関連するシステムメモリに格納されたデータは、アクティブなままであり、実行可能なコードの場合には、実行されることになる。処理のクラウドランタイムプラットフォーム関連リソースの少なくとも一部が、解放（released）または解放（freed-up）されてもよいことに留意されたい。すべてのリソースが自動的に解放されることは要求されない。

【0089】

いくつかの実施形態では、方法100は、処理のハイバネーション中に、非同期操作が実行される操作108に進む。非同期操作は、例えば、異なる地理的位置（例えば、典型的には、異なるクラウドコンピューティングセンター）に位置する別のクラスタの別のノード上で潜在的にシステムコールであってよい。

【0090】

いくつかの実施形態では、方法100は、操作110に進む。操作110では、非同期操作の完了後、処理に対して開始された補完関数、または補完関数呼び出しが遮断される。いくつかの実施形態では、方法100は、操作112に進む。操作112において、遮断は、呼び出し元プログラムにも返されるように、補完関数に追加プログラムコードまたはデータ、またはその両方を注入するために使用され得る。

【0091】

いくつかの実施形態において、方法100は、操作114に進む。操作114において、処理はハイバネーション解除され、処理の解放されたクラウドランタイムプラットフォーム関連リソースが再割り当てされる。いくつかの実施形態では、方法100は、操作116に進む。操作116において、非同期操作の結果データと、処理に対する追加プログラムコードとデータの少なくとも1つとを返す補完関数が実行され、それによって、サーバレスクラウドランタイムプラットフォーム上でのシングルスレッド、非同期I/Oベースのプログラムコードの実行が可能になる。このようにして、有利には「古い」シングルスレッドのプログラムコード、特に長いトランザクションのためのプログラムコードを、ステートレスクラウドコンピューティングのフレームワークにおいて何の変更もなく操作することができる。

【0092】

ここで、関連するコンポーネントおよびそれらの間の相互作用の実施形態200のブロック図を示す図2を参照する。左内側で、ブロック202は、開始と終了を有するシングルスレッドプログラムコードを表している。しかしながら、それはまた、ユーザ入力によってのみ、またはオペレーティングシステムベースのサービスによってのみ終了され得るループ化されたプログラムであってもよい。

【0093】

プログラム202の操作中に配置ルーチン208の支援を借りて非同期I/O操作204のような非同期操作が開始される。非同期操作の実行を制御するために、いくつかのパラメータ値が調整されることがある。シングルスレッドプログラムコード202に関連するオペレーティングシステムプロセス（不図示）は、システムメモリ（例えば、関連RAM）とは別に、非同期操作204の実行中にハイバネーションされてもよい。

10

20

30

40

50

【0094】

非同期操作204の終了時に、例えば別のシステムコールの形で、補完関数206が実行されてもよい。しかし、非同期操作204の結果は、シングルスレッドプログラムコード202、およびそれぞれの処理に直接返されない。その代わりに、他のクロス操作204からの結果データをプログラムコード202に返そうとするときに補完関数が中断され、追加プログラムコードもしくはデータまたはその両方210が戻り経路212に注入される。

【0095】

このようにして、シングルスレッドプログラムコード202をハイバネーション解除する前に、あるいは関連するオペレーティングシステム処理をより向上させるために、シングルスレッドプログラムコードがその実行を継続する前に、追加のファンクションポイントを注入することが可能である。

10

【0096】

ここで、サーバレスクラウドランタイムプラットフォーム上でシングルスレッド、非同期I/Oベースのプログラムコードを実行するためのランタイムシステム300の一実施形態のブロック図を示す図3を参照する。ランタイムシステムは、プログラムコードの非同期操作を実行するために適合された第1の実行モジュール302を含む。それによって、非同期操作は、補完関数で完了されることになる。ランタイムシステムはまた、非同期操作に関連する処理をハイバネーションするために適合されたハイバネーションモジュール304と、関連するシステムメモリを除く処理のクラウドランタイムプラットフォーム関連リソースを解放するために適合された解放ユニット306とを含む。

20

【0097】

さらに、ランタイムシステム300は、処理のハイバネーション中に、非同期操作を実行するために適合された第2の実行モジュール308と、非同期操作の完了後、処理に対して開始された補完関数を遮断するために適合されたインターセプタ310と、補完関数に追加プログラムコードとデータの少なくとも1つを注入するために適合されたインジェクタモジュール312とを含む。

【0098】

ランタイムシステム300は、処理をハイバネーション解除し、処理のクラウドランタイムプラットフォーム関連リソースを再割り当て、解放(freeing-up)または解放(releasing)するために適合されたハイバネーション解除モジュール314も含む。ランタイムシステム300はまた、非同期操作の結果データと、処理に対する追加プログラムコードとデータの少なくとも1つとを返す補完関数を実行するために適合された第3の実行モジュール316を含み、それによってシングルスレッド、非同期I/Oの実行が有効化される。

30

【0099】

ランタイムシステム300のユニットおよびモジュールは、信号交換のために通信可能に結合されることに留意されたい。したがって、第1の実行モジュール302、ハイバネーションモジュール304、解放ユニット306、第2の実行モジュール308、インターセプタ310、インジェクタモジュール312、ハイバネーション解除モジュール314、および第3の実行モジュール316は、そのような方法で互いにリンクされてもよい。あるいは、これらのモジュールにおけるユニットは、ランタイムシステム内部バスシステム318に接続されてもよい。

40

【0100】

ここで、クラウドコンピューティング環境410を示す図4Aを参照する。図示されるように、クラウドコンピューティング環境410は、クラウド消費者が使用するローカルコンピュータ装置(例えば、パーソナルデジタルアシスタント(PDA)または携帯電話400A、デスクトップコンピュータ400B、ラップトップコンピュータ400C、もしくは自動車コンピュータシステム400Nまたはこれらの組み合わせなど)が通信する1または複数のクラウドコンピューティングノード400を含んでいる。ノード400は

50

互いに通信することができる。それらは、例えば、上述のプライベート、コミュニティ、パブリックもしくはハイブリッドクラウドまたはこれらの組み合わせなど、1つまたは複数のネットワークにおいて、物理的または仮想的にグループ化（不図示）することができる。

【0101】

これにより、クラウドコンピューティング環境410は、サービスとしてのインフラストラクチャ、プラットフォームもしくはソフトウェアまたはこれらの組み合わせを提供することができる。クラウド消費者はこれらについて、ローカルコンピュータ装置上にリソースを維持する必要がない。なお、図4Aに示すコンピュータ装置400A～Nの種類は例示に過ぎず、コンピューティングノード400およびクラウドコンピューティング環境410は、任意の種類のネットワークもしくはネットワークアドレス指定可能接続（例えば、ウェブブラウザの使用）またはその両方を介して、任意の種類の電子装置と通信可能であることを理解されたい。

【0102】

図4Bは、クラウドコンピューティング環境410（図4A）によって提供される機能的抽象化レイヤのセットを示す。なお、図4Bに示すコンポーネント、レイヤおよび機能は例示に過ぎず、本開示の実施形態はこれらに限定されないことをあらかじめ理解されたい。図示するように、以下のレイヤおよび対応する機能が提供される。ハードウェアおよびソフトウェアレイヤ415は、ハードウェアコンポーネントおよびソフトウェアコンポーネントを含む。ハードウェアコンポーネントの例には、メインフレーム402、サーバ406、RISC（縮小命令セットコンピュータ）アーキテクチャベースのサーバ404、ブレードサーバ408、記憶装置411、ならびにネットワークおよびネットワークコンポーネント412が含まれる。いくつかの実施形態において、ソフトウェアコンポーネントは、ネットワークアプリケーションサーバソフトウェア414およびデータベースソフトウェア416またはその両方を含む。

【0103】

仮想化レイヤ420は、抽象化レイヤを提供する。当該レイヤから、例えば以下の仮想エンティティを提供することができる：仮想サーバ422、仮想ストレージ424、仮想プライベートネットワークを含む仮想ネットワーク426、仮想アプリケーションおよびオペレーティングシステム428、ならびに仮想クライアント430。

【0104】

一例として、管理レイヤ440は以下の機能を提供することができる。リソース準備442は、クラウドコンピューティング環境内でタスクを実行するために利用されるコンピューティングリソースおよび他のリソースの動的な調達を提供する。計量および価格設定444は、クラウドコンピューティング環境内でリソースが利用される際のコスト追跡、およびこれらのリソースの消費に対する請求またはインボイス送付を提供する。一例として、これらのリソースはアプリケーションソフトウェアのライセンスを含んでよい。セキュリティは、データおよび他のリソースに対する保護のみならず、クラウドコンシューマおよびタスクの識別確認を提供する。ユーザポータル446は、コンシューマおよびシステム管理者にクラウドコンピューティング環境へのアクセスを提供する。サービスレベル管理448は、要求されたサービスレベルが満たされるように、クラウドコンピューティングリソースの割り当ておよび管理を提供する。サービス品質保証（SLA）の計画および履行450は、SLAに従って将来必要になると予想されるクラウドコンピューティングリソースの事前手配および調達を提供する。

【0105】

ワークロードレイヤ460は、クラウドコンピューティング環境が利用可能な機能の例を提供する。このレイヤから提供可能なワークロードおよび機能の例には、マッピングおよびナビゲーション462、ソフトウェア開発およびライフサイクル管理464、仮想教室教育の配信466、データ分析処理468、取引処理470、ならびに、ランタイムシステム472（図3を対比させると、ランタイムシステム300）が含まれる。

【0106】

本開示の実施形態は、プログラムコードの格納もしくは実行またはその両方に適したプラットフォームであるかにかかわらず、実質的にあらゆるタイプのコンピュータとともに実装することができる。

【0107】

ここで、本開示の実施形態に従って、本明細書に記載される方法、ツール、およびモジュール、ならびに任意の関連機能の1または複数を実装する際に（例えば、コンピュータの1または複数のプロセッサ回路またはコンピュータプロセッサを使用して）使用され得る別の例示的コンピュータシステム500のハイレベルブロック図を示す図5を参照する。いくつかの実施形態では、コンピュータシステム500の主要コンポーネントは、1または複数のCPU502、メモリサブシステム504、端末インタフェース512、ストレージインタフェース516、I/O（入力/出力）デバイスインタフェース514、およびネットワークインタフェース518を含んでもよく、これらのすべては、メモリバス503、I/Oバス508、およびI/Oバスインタフェースユニット510を介してコンポーネント間の通信のために直接的または間接的に通信可能に結合され得る。

【0108】

コンピュータシステム500は、1または複数の汎用プログラマブル中央処理装置（CPU）502A、502B、502C、および502D（ここでは一般的にCPU502と称する）を含んでもよい。いくつかの実施形態では、コンピュータシステム500は、比較的大規模なシステムに典型的な複数のプロセッサを含んでもよいが、他の実施形態では、コンピュータシステム500は、代替的に、単一のCPUシステムであってもよい。各CPU502は、メモリサブシステム504に格納された命令を実行してもよく、1または複数のレベルのオンボードキャッシュを含んでもよい。

【0109】

システムメモリ504は、ランダムアクセスメモリ（RAM）522またはキャッシュメモリ524などの揮発性メモリの形態のコンピュータシステム読取可能媒体を含んでもよい。コンピュータシステム501は、他の取り外し可能/取り外し不可能な、揮発性/不揮発性のコンピュータシステム記憶媒体をさらに含むことができる。単なる例示だが、ストレージシステム526は、「ハードドライブ」などの非リムーバブル、不揮発性磁気媒体からの読み取りおよびそれに対する書き込みのために提供され得る。図示しないが、取り外し可能な不揮発性磁気ディスク（例えば、「フロッピーディスク」）からの読み取りおよびそれに対する書き込みのための磁気ディスクドライブ、またはCD-ROM、DVD-ROMまたは他の光学媒体などの取り外し可能な不揮発性光ディスクからの読み取りまたは書き込みのための光ディスクドライブを提供することができる。さらに、メモリ504は、フラッシュメモリ（例えば、フラッシュメモリスティックドライブまたはフラッシュドライブ）を含むことができる。メモリデバイスは、1または複数のデータメディアインタフェースによってメモリバス503に接続することができる。メモリ504は、様々な実施形態の機能を実行するように構成されたプログラムモジュールのセット（例えば、少なくとも1つ）を有する少なくとも1つのプログラム製品を含むことができる。

【0110】

それぞれが少なくとも1組のプログラムモジュール530を有する1または複数のプログラム/ユーティリティ528は、メモリ504に格納されてもよい。プログラム/ユーティリティ528は、ハイパーバイザー（仮想マシンモニタとも呼ばれる）、1または複数のオペレーティングシステム、1または複数のアプリケーションプログラム、他のプログラムモジュール、およびプログラムデータを含んでもよい。オペレーティングシステム、1または複数のアプリケーションプログラム、他のプログラムモジュール、およびプログラムデータの各々またはそれらの一部の組み合わせは、ネットワーキング環境の実装を含んでもよい。プログラム528もしくはプログラムモジュール530またはその両方は、一般に、様々な実施形態の機能または方法論を実行する。

【0111】

メモリバス503は、図5では、CPU502、メモリサブシステム504、およびI/Oバスインタフェース510の間の直接通信経路を提供する単一のバス構造として示されているが、メモリバス503は、いくつかの実施形態では、階層的、スターまたはウェブ構成、複数の階層的なバス、並列および冗長経路、または他の適切なタイプの構成におけるポイントツーポイントリンクなど様々な形態のいずれで配置してもよい複数の異なるバスまたは通信経路を含んでもよい。さらに、I/Oバスインタフェース510およびI/Oバス508は、単一のそれぞれのユニットとして示されているが、コンピュータシステム500は、いくつかの実施形態において、複数のI/Oバスインタフェースユニット510、複数のI/Oバス508、または両方を含んでもよい。さらに、I/Oバス508を様々なI/Oデバイスに走る様々な通信経路から分離する複数のI/Oインタフェースユニットが示されているが、他の実施形態では、I/Oデバイスの一部または全部が1または複数のシステムI/Oバスに直接接続されてもよい。

10

【0112】

いくつかの実施形態では、コンピュータシステム500は、マルチユーザメインフレームコンピュータシステム、シングルユーザシステム、または、直接的なユーザインタフェースをほとんどまたは全く有さないが、他のコンピュータシステム（クライアント）からの要求を受信するサーバコンピュータまたは同様のデバイスであってもよい。さらに、いくつかの実施形態では、コンピュータシステム500は、デスクトップコンピュータ、ポータブルコンピュータ、ラップトップまたはノートブックコンピュータ、タブレットコンピュータ、ポケットコンピュータ、電話、スマートフォン、ネットワークスイッチまたはルータ、または任意の他の適切なタイプの電子デバイスとして実装されてもよい。

20

【0113】

図5は、例示的なコンピュータシステム500の代表的な主要コンポーネントを図示することを意図していることに留意されたい。しかしながら、いくつかの実施形態では、個々のコンポーネントは、図5に表されたものよりも大きいまたは小さい複雑さを有してよく、図5に示されたもの以外のまたはこれに加えてコンポーネントが存在してもよく、そのようなコンポーネントの数、タイプ、および構成は異なってもよい。

【0114】

本明細書でより詳細に議論するように、本明細書に記載される方法の実施形態のいくつかの操作の一部または全部は、代替の順序で実行されてもよく、または全く実行されなくてもよいことが企図される。さらに、複数の操作は同時にまたはより大きな処理の内部部分として発生してもよい。

30

【0115】

本開示の様々な実施形態の説明は、例示の目的で提示されているが、網羅的であることを意図するものではなく、開示される実施形態に限定されることを意図するものでもない。記載される実施形態の範囲および精神から逸脱することなく、多くの修正および変更が可能であることは当業者には明らかであろう。本明細書で使用する用語は、実施形態の原理、市場で見られる技術に対する実際の適用または技術的改善を最もよく説明するため、または当業者が本明細書に記載の実施形態を理解できるようにするために選択された。

【0116】

本開示は、任意の可能な技術詳細レベルで統合されたシステム、方法もしくはコンピュータプログラム製品またはそれらの組み合わせとすることができる。コンピュータプログラム製品は、プロセッサに本開示の態様を実行させるためのコンピュータ可読プログラム命令を記憶したコンピュータ可読記憶媒体を含んでもよい。

40

【0117】

媒体は、伝播媒体のための電子、磁気、光学、電磁気、赤外線、または半導体システムであってもよい。コンピュータ可読媒体の例としては、半導体または固体メモリ、磁気テープ、取り外し可能なコンピュータディスク、ランダムアクセスメモリ（RAM）、読取専用メモリ（ROM）、硬質磁気ディスクおよび光ディスクを挙げることができる。光ディスクの現在の例としては、コンパクトディスク読取専用メモリ（CD-ROM）、

50

コンパクトディスクリード／ライト（ＣＤ－Ｒ／Ｗ）、ＤＶＤおよびブルーレイディスクが挙げられる。

【０１１８】

コンピュータ可読記憶媒体は、使用するために命令実行装置によって命令を保持および記憶する有形の装置である。コンピュータ可読記憶媒体は、例えば、これらに限定されないが、電子記憶装置、磁気記憶装置、光記憶装置、電磁記憶装置、半導体記憶装置、または前述を任意に適宜組み合わせたものであってよい。コンピュータ可読記憶媒体のより具体的な一例を列挙すると、ポータブルコンピュータディスク、ハードディスク、ランダムアクセスメモリ（ＲＡＭ）、読み取り専用メモリ（ＲＯＭ）、書き込みおよび消去可能なランダムアクセスメモリ（ＥＰＲＯＭまたはフラッシュメモリ）、静的ランダムアクセスメモリ（ＳＲＡＭ）、ポータブルコンパクトディスク読み取り専用メモリ（ＣＤ－ＲＯＭ）、デジタル多用途ディスク（ＤＶＤ）、メモリスティック、フロッピーディスク、機械的命令が記録された溝を持つパンチカードまたは隆起構造などの符号化されたデバイス、および前述を任意に適宜組み合わせたものである。本明細書で使用されるコンピュータ可読記憶媒体は、電波または他の自由に伝播する電磁波、導波管または他の伝送媒体を通して伝播する電磁波（例えば、光ファイバーケーブルを通過する光パルス）、またはワイヤーを介して送信される電気信号などの一時的な信号自体であると解釈されるべきではない。

【０１１９】

本明細書に記載のコンピュータ可読プログラム命令は、コンピュータ可読記憶媒体からそれぞれのコンピューティング／処理装置に、または、ネットワーク（例えば、インターネット、ローカルエリアネットワーク、ワイドエリアネットワーク、もしくはワイヤレスネットワークネットワークまたはその組み合わせ）を介して外部コンピュータまたは外部記憶装置にダウンロードすることができる。ネットワークは、銅線伝送ケーブル、光伝送ファイバー、無線伝送、ルーター、ファイアウォール、スイッチ、ゲートウェイコンピュータ、もしくはエッジサーバーまたはその組み合わせで構成されることができる。各コンピューティング／処理装置のネットワークアダプタカードまたはネットワークインターフェースは、ネットワークからコンピュータ可読プログラム命令を受信し、それぞれのコンピューティング／処理装置内のコンピュータ可読記憶媒体に格納するためにコンピュータ可読プログラム命令を転送する。

【０１２０】

本開示の動作を実行するためのコンピュータ可読プログラム命令は、アセンブラ命令、命令セットアーキテクチャ（ＩＳＡ）命令、機械命令、機械依存命令、マイクロコード、ファームウェア命令、状態設定データ、統合回路のための構成データ、またはSmalltalk、Ｃ＋＋などのオブジェクト指向プログラミング言語と「Ｃ」プログラミング言語や類似のプログラミング言語などの手続き型プログラミング言語を含む、１つ以上のプログラミング言語の任意の組み合わせで記述されたソースコードまたはオブジェクトコードのいずれかであってよい。コンピュータ可読プログラム命令は、スタンドアロンソフトウェアパッケージとして、完全にユーザのコンピュータ上で、または部分的にユーザのコンピュータ上で実行可能である。あるいは、部分的にユーザのコンピュータ上でかつ部分的にリモートコンピュータ上で、または完全にリモートコンピュータまたはサーバ上で実行可能である。後者のシナリオでは、リモートコンピュータは、ローカルエリアネットワーク（ＬＡＮ）またはワイドエリアネットワーク（ＷＡＮ）を含む任意のタイプのネットワークを介してユーザのコンピュータに接続され、または（例えば、インターネットサービスプロバイダーを使用したインターネット経由で）外部コンピュータに接続されてよい。いくつかの実施形態では、例えば、プログラマブルロジック回路、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、またはプログラマブルロジックアレイ（ＰＬＡ）を含む電子回路は、本開示の態様を実行するために、コンピュータ可読プログラム命令の状態情報を利用してパーソナライズすることにより、コンピュータ可読プログラム命令を実行することができる。

10

20

30

40

50

【0121】

本開示の態様は、本開示の実施形態による方法、装置（システム）、およびコンピュータプログラム製品のフローチャート図もしくはブロック図またはその両方を参照して本明細書に記載されている。フローチャート図もしくはブロック図またはその両方の各ブロック、およびフローチャート図もしくはブロック図またはその両方のブロックの組み合わせは、コンピュータ可読プログラム命令によって実装できることが理解されよう。

【0122】

これらのコンピュータ可読プログラム命令は、コンピュータまたは他のプログラム可能なデータ処理装置のプロセッサを介して実行される命令がフローチャートもしくはブロック図またはその両方の1つまたは複数のブロックで指定された機能／動作を実装するための手段を生成するように、機械を生成するために汎用コンピュータまたは専用コンピュータのプロセッサまたは他のプログラム可能なデータ処理装置に提供されることができる。これらのコンピュータ可読記憶媒体はまた、フローチャートもしくはブロック図またはその両方の1つまたは複数のブロックで指定された機能／行為の態様を実装する命令を含む生成品の1つを命令が記憶されたコンピュータ可読プログラム命令が構成するように、コンピュータ、プログラム可能なデータ処理装置、もしくは特定の 방법으로機能する他のデバイスまたはその組み合わせに接続可能なコンピュータ可読記憶媒体の中に記憶されることができる。

【0123】

コンピュータ、他のプログラム可能な装置、または他のデバイス上でフローチャートもしくはブロック図またはその両方の1つまたは複数のブロックで指定された機能／行為を実行する命令のように、コンピュータ可読プログラム命令はまた、コンピュータ、他のプログラム可能な装置、または他のデバイスにロードされ、コンピュータ、他のプログラム可能な装置、または他のデバイス上で一連の操作ステップを実行し、コンピュータ実装された過程を生成することができる。

【0124】

図中のフローチャートおよびブロック図またはその両方は、本開示の様々な実施形態によるシステム、方法、およびコンピュータプログラム製品が実装可能な構成、機能、および動作を示している。これに関して、フローチャートまたはブロック図の各ブロックは、モジュール、セグメント、または命令の一部を表してよく、これは、指定された論理機能を実装するための1つまたは複数の実行可能命令を構成する。いくつかの代替の実施形態では、ブロックに示されている機能は、図に示されている順序とは異なる場合がある。例えば、連続して示される2つのブロックは、実際には、1つのステップとして達成される場合があり、同時に、実質的に同時に、部分的または全体的に時間的に重複する方法で実行されるか、またはブロックは、関連する機能に応じて逆の順序で実行される場合がある。ブロック図もしくはフローチャート図またはその両方の各ブロック、およびブロック図もしくはフローチャート図またはその両方のブロックの組み合わせは、指定された機能または動作を実行する、または特別な目的のハードウェアとコンピュータ命令の組み合わせを実行する特別な目的のハードウェアベースのシステムによって実装できることに留意されたい。本明細書で用いる用語は、特定の実施形態を説明するためのものに過ぎず、本開示を限定することを意図したものではない。本明細書で使用される場合、単数形「ある／1つの／一の (a)」、「ある／1つの／一の (an)」および「その／当該 (the)」は、文脈からそうではないことが明らかでない限り、複数形を含むことが意図される。さらに、本明細書で使用される場合、「含む (comprises)」もしくは「含む (comprising)」またはその両方の用語は、記載された特徴、整数、工程、動作、要素もしくはコンポーネントまたはこれらの組み合わせの存在を特定するものであるが、1つ以上の他の特徴、整数、工程、動作、要素、コンポーネントもしくはそれらのグループまたはこれらの組み合わせが存在したり、追加されたりすることを排除するものではないことが理解されるであろう。以下の特許請求の範囲における全ての手段またはステップに機能要素を加えた対応する構造、材料、行為、および等価物は、具体的に請求されるように、他の請求され

10

20

30

40

50

る要素と組み合わせて機能を実行するための任意の構造、材料、または行為を含むことを意図している。本開示の説明は、例示および説明のために提示されたが、開示された形態での開示を網羅または限定することを意図していない。本開示の範囲および精神から逸脱することなく、多くの修正および変更が可能であることは当業者には明らかであろう。本開示の原理および実用化を最もよく説明するために、また、当業者の他の者が、企図される特定の用途に適するように、様々な変更を伴う種々の実施形態について本開示を理解できるように、実施形態が選択および説明されている。

【 0 1 2 5 】

本開示は、特定の実施形態の観点から説明されてきたが、当業者には、その変更および修正が明らかになることが予想される。したがって、以下の特許請求の範囲は、本開示の真の精神および範囲内に入るようなすべての変更および修正をカバーするものとして解釈されることが意図される。

10

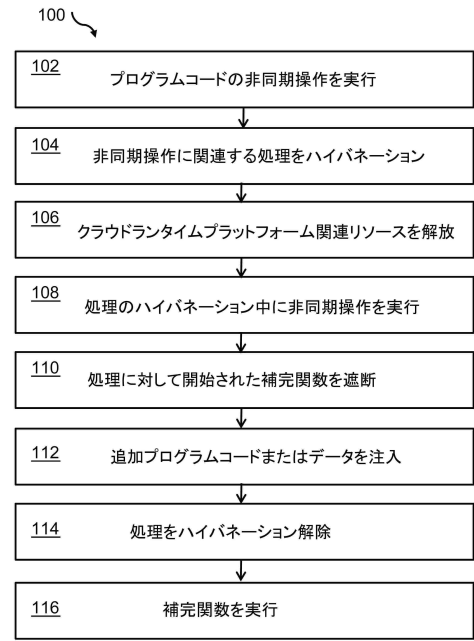
20

30

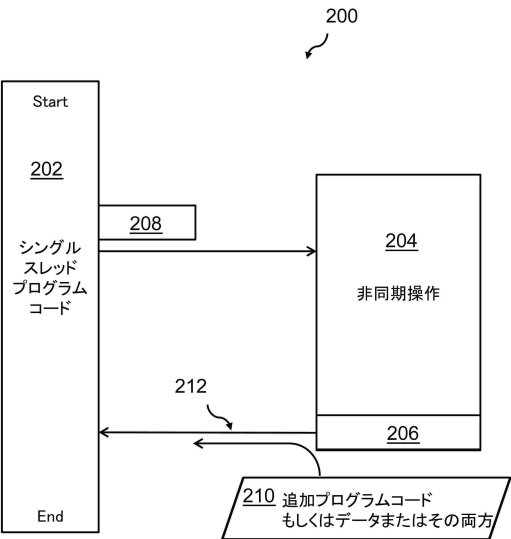
40

50

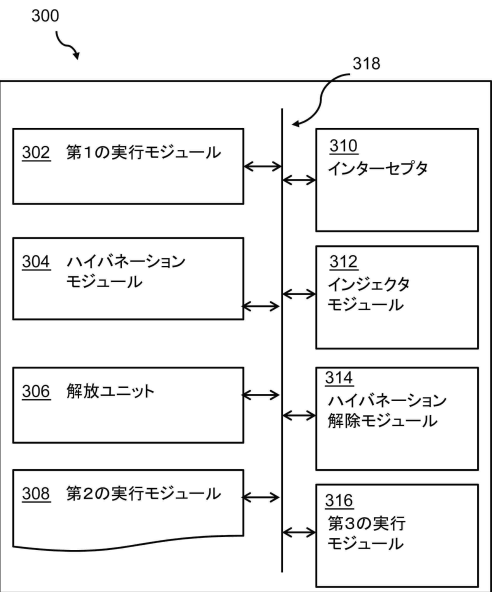
【図面】
【図 1】



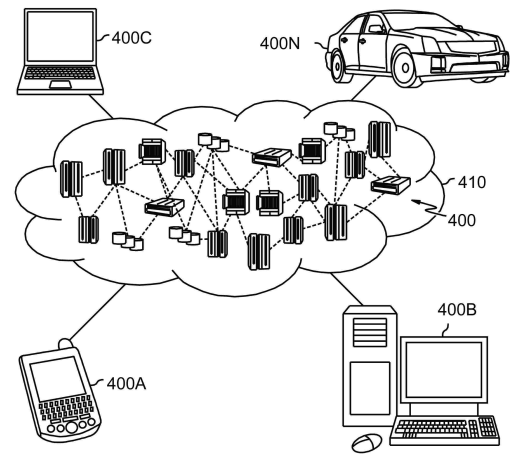
【図 2】



【図 3】



【図 4 A】



10

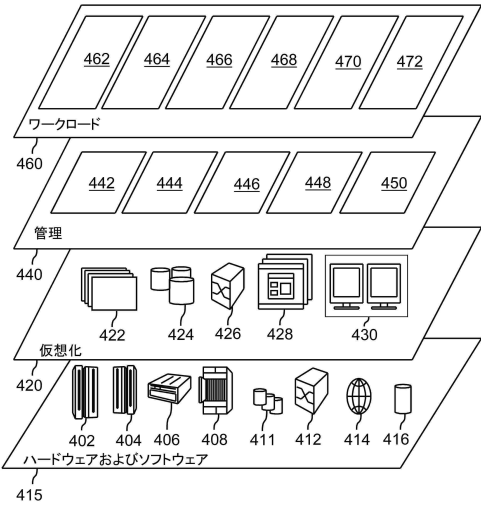
20

30

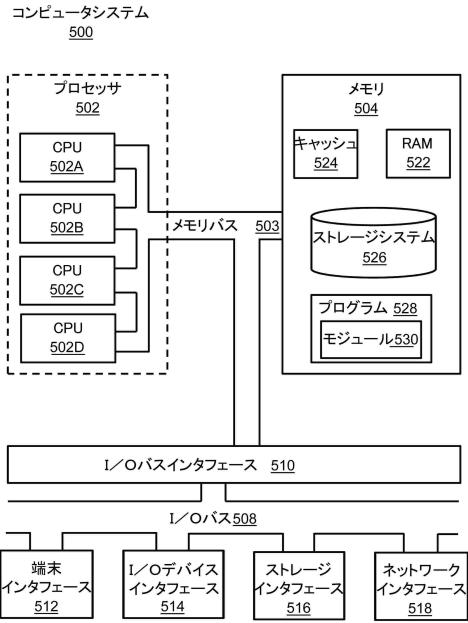
40

50

【図 4 B】



【図 5】



10

20

30

40

50

フロントページの続き

- (74)代理人 100120710
弁理士 片岡 忠彦
- (72)発明者 タイヒ、トルステン
ドイツ連邦共和国 7 1 0 3 2 ベー布林ゲン シェーナハイチャー・シュトラーセ 2 2 0 アイ・ビー・エム・ドイツ・リサーチ・アンド・デベロップメント・ジー・エム・ビー・エイチ内
- (72)発明者 アグラワル、スガンダ
ドイツ連邦共和国 7 1 0 3 2 ベー布林ゲン シェーナハイチャー・シュトラーセ 2 2 0 アイ・ビー・エム・ドイツ・リサーチ・アンド・デベロップメント・ジー・エム・ビー・エイチ内
- (72)発明者 ピットナー、ダニエル
ドイツ連邦共和国 7 1 0 3 2 ベー布林ゲン シェーナハイチャー・シュトラーセ 2 2 0 アイ・ビー・エム・ドイツ・リサーチ・アンド・デベロップメント・ジー・エム・ビー・エイチ内
- (72)発明者 シュウィンンド、マーク
ドイツ連邦共和国 7 1 0 3 2 ベー布林ゲン シェーナハイチャー・シュトラーセ 2 2 0 アイ・ビー・エム・ドイツ・リサーチ・アンド・デベロップメント・ジー・エム・ビー・エイチ内
- 審査官 松平 英
- (56)参考文献 特開 2 0 1 7 - 0 3 3 3 3 1 (J P , A)
特開 2 0 0 7 - 0 2 6 0 0 8 (J P , A)
特開 2 0 1 8 - 0 7 3 4 0 5 (J P , A)
- (58)調査した分野 (Int.Cl., D B 名)
G 0 6 F 8 / 0 0 - 8 / 3 8
8 / 6 0 - 8 / 7 7
9 / 4 4 - 9 / 4 4 5
9 / 4 5 1
9 / 4 5 5 - 9 / 5 4