

(19)日本国特許庁(JP)

(12)公表特許公報(A)

(11)公表番号
特表2024-529878
(P2024-529878A)

(43)公表日 令和6年8月14日(2024.8.14)

(51)国際特許分類

F I

テーマコード (参考)

G 0 6 F 8/71 (2018.01) G 0 6 F 8/71 5 B 0 4 2

G 0 6 F 11/36 (2006.01) G 0 6 F 11/36 1 2 4 5 B 3 7 6

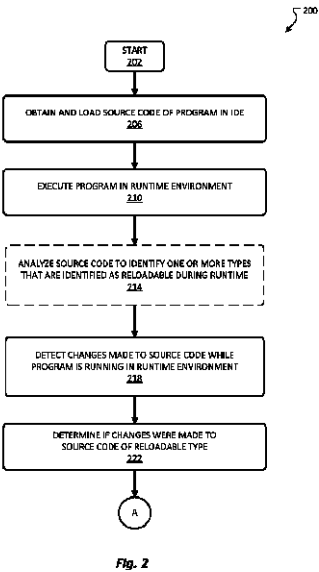
審査請求 未請求 予備審査請求 未請求 (全26頁)

(21)出願番号	特願2024-501863(P2024-501863)	(71)出願人	314015767
(86)(22)出願日	令和4年6月29日(2022.6.29)		マイクロソフト テクノロジー ライセン
(85)翻訳文提出日	令和6年3月8日(2024.3.8)		シング, エルエルシー
(86)国際出願番号	PCT/US2022/035398		アメリカ合衆国 ワシントン州 9 8 0 5
(87)国際公開番号	WO2023/287574		2 レッドモンド ワン マイクロソフト
(87)国際公開日	令和5年1月19日(2023.1.19)		ウェイ
(31)優先権主張番号	17/373,391	(74)代理人	100118902
(32)優先日	令和3年7月12日(2021.7.12)		弁理士 山本 修
(33)優先権主張国・地域又は機関		(74)代理人	100106208
	米国(US)		弁理士 宮前 徹
(81)指定国・地域	AP(BW,GH,GM,KE,LR,LS,MW,MZ,NA	(74)代理人	100196508
	,RW,SD,SL,ST,SZ,TZ,UG,ZM,ZW),EA(		弁理士 松尾 淳一
	AM,AZ,BY,KG,KZ,RU,TJ,TM),EP(AL,A	(72)発明者	マトウゼク, トーマス
	T,BE,BG,CH,CY,CZ,DE,DK,EE,ES,FI,FR		アメリカ合衆国 ワシントン州 9 8 0 5
	,GB,GR,HR,HU,IE,IS,IT,LT,LU,LV,MC,		2-6399 レッドモンド ワン マイ
	最終頁に続く		最終頁に続く

(54)【発明の名称】 ランタイムにリロード可能タイプのソースコードに対して行われた変更を実施すること

(57)【要約】

ランタイムにプログラムのソースコードに対して行われた変更を実施するためのシステムおよび方法が提供される。具体的には、コンピューティングデバイスは、ランタイム環境でソースコードを実行し、ランタイム環境でランタイムにソースコードに対して行われた変更を検出し得る。変更を検出したことに応答して、コンピューティングデバイスは、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定し得る。リロード可能と識別されたソースコード内のタイプに対して変更が行われたと判定したことに応答して、コンピューティングデバイスは、タイプの新しいバージョンを作成し、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行い得る。



【特許請求の範囲】

【請求項 1】

ランタイムにプログラムのソースコードに対して行われた変更を実施するためのコンピューティングデバイスであって、

プロセッサと、

複数の命令が記憶されたメモリと

を備え、前記複数の命令は、前記プロセッサによって実行されると、前記コンピューティングデバイスに、

ランタイム環境で前記ソースコードを実行することと、

前記ランタイム環境でランタイムに前記ソースコードに対して行われた変更を検出することと、

リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたかどうかを判定することと、

リロード可能と識別された前記ソースコード内の前記タイプに対して前記変更が行われたという判定に応答して、前記タイプの新しいバージョンを作成することと、

前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うことと

をさせる、

コンピューティングデバイス。

【請求項 2】

請求項 1 に記載のコンピューティングデバイスであって、ランタイム環境で前記ソースコードを実行することは、

前記ソースコードを取得することと、

統合開発環境（IDE）内に前記ソースコードをロードすることと、

前記 IDE によって配置された前記ランタイム環境で前記ソースコードを実行することと

を含む、コンピューティングデバイス。

【請求項 3】

請求項 1 に記載のコンピューティングデバイスであって、リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われなかったという判定に応答して、前記タイプをインプレースで更新することと、前記ランタイム環境内の前記変更を伴う更新後の前記ソースコードの実行を続行することとの一方または双方を行うようにさらに構成される、コンピューティングデバイス。

【請求項 4】

請求項 1 に記載のコンピューティングデバイスであって、

前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うことは、前記ソースコード内の前記タイプの前のバージョンのランタイム表現および 1 つまたは複数の既存のインスタンスを、前記タイプの前記新しいバージョンに置き換えることを含むことと、

リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたかどうかを判定することは、前記ソースコードを構築するために使用されたフレームワークに関連付けられるフレームワークライブラリに基づいて、リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたかどうかを判定することをさらに含み、前記フレームワークライブラリは、リロード可能とマークされる事前定義されたタイプのリストを含むことと

の一方または双方である、

コンピューティングデバイス。

【請求項 5】

請求項 1 に記載のコンピューティングデバイスであって、前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うことは、

前記ソースコードを構築するために使用されたフレームワークに、前記タイプの前記新しいバージョンが作成されたことを通知することと、

前記ソースコード内の前記タイプの前バージョンの1つまたは複数の既存のインスタンスから前記新しいバージョンに状態を移すこととを含む、コンピューティングデバイス。

【請求項6】

ランタイムにプログラムのソースコードに対して行われた変更を実施するための命令を記憶する非一時的コンピュータ可読媒体であって、前記命令は、コンピューティングデバイスの1つまたは複数のプロセッサによって実行されると、前記コンピューティングデバイスに、

ランタイム環境で前記ソースコードを実行することと、

前記ランタイム環境でランタイムに前記ソースコードに対して行われた変更を検出することと、

リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたかどうかを判定することと、

リロード可能と識別された前記ソースコード内の前記タイプに対して前記変更が行われたという判定に応答して、前記タイプの新しいバージョンを作成することと、

前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うことと

をさせる、非一時的コンピュータ可読媒体。

【請求項7】

請求項6に記載の非一時的コンピュータ可読媒体であって、ランタイム環境で前記ソースコードを実行することは、

前記ソースコードを取得することと、

統合開発環境（IDE）内に前記ソースコードをロードすることと、

前記IDEによって配置された前記ランタイム環境で前記ソースコードを実行することと

を含む、非一時的コンピュータ可読媒体。

【請求項8】

請求項6に記載の非一時的コンピュータ可読媒体であって、前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うことは、前記ソースコード内の前記タイプの前バージョンのランタイム表現および1つまたは複数の既存のインスタンスを、前記タイプの前記新しいバージョンに置き換えることを含む、非一時的コンピュータ可読媒体。

【請求項9】

請求項6に記載の非一時的コンピュータ可読媒体であって、前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うことは、

前記ソースコードを構築するために使用されたフレームワークに、前記タイプの前記新しいバージョンが作成されたことを通知することと、

前記ソースコード内の前記タイプの前バージョンの1つまたは複数の既存のインスタンスから前記新しいバージョンに状態を移すことと

を含む、非一時的コンピュータ可読媒体。

【請求項10】

ランタイムにプログラムのソースコードに対して行われた変更を実施するための方法であって、

ランタイム環境で前記ソースコードを実行するステップと、

前記ランタイム環境でランタイムに前記ソースコードに対して行われた変更を検出するステップと、

リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたか

10

20

30

40

50

どうかを判定するステップと、

リロード可能と識別された前記ソースコード内の前記タイプに対して前記変更が行われたと判定したことに応答して、前記タイプの新しいバージョンを作成するステップと、

前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うステップと

を含む方法。

【請求項 1 1】

請求項 1 0 に記載の方法であって、ランタイム環境で前記ソースコードを実行するステップは、

前記ソースコードを取得するステップと、

統合開発環境（IDE）内に前記ソースコードをロードするステップと、

前記 IDE によって配置された前記ランタイム環境で前記ソースコードを実行するステップと

を含む、方法。

【請求項 1 2】

請求項 1 0 に記載の方法であって、リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われなかったと判定したことに応答して、前記タイプをインプレースで更新するステップと、前記ランタイム環境内の前記変更を伴う更新後のソースコードの実行を続行するステップとの一方または双方をさらに含む方法。

【請求項 1 3】

請求項 1 0 に記載の方法であって、

前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うステップは、前記ソースコード内の前記タイプの前のバージョンのランタイム表現および 1 つまたは複数の既存のインスタンスを、前記タイプの前記新しいバージョンに置き換えるステップをさらに含む、方法。

【請求項 1 4】

請求項 1 0 に記載の方法であって、前記タイプの前記新しいバージョンを使用して、ランタイムに前記ソースコードに対する前記変更の実施を行うステップは、

前記ソースコードを構築するために使用されたフレームワークに、前記タイプの前記新しいバージョンが作成されたことを通知するステップと、

前記ソースコード内の前記タイプの前のバージョンの 1 つまたは複数の既存のインスタンスから前記新しいバージョンに状態を移すステップと

をさらに含む、方法。

【請求項 1 5】

請求項 1 0 に記載の方法であって、リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたかどうかを判定するステップは、前記ソースコードを構築するために使用されたフレームワークに関連付けられるフレームワークライブラリに基づいて、リロード可能と識別された前記ソースコード内のタイプに対して前記変更が行われたかどうかを判定するステップをさらに含み、前記フレームワークライブラリは、リロード可能とマークされる事前定義されたタイプのリストを含む、方法。

【発明の詳細な説明】

【背景技術】

【0001】

様々なタイプのソフトウェア開発ツールは、プログラムがデバッグセッション中にブレークモードにある間、ソフトウェア開発者がアプリケーションのソースコードに対して変更を行うことを可能にする。デバッグセッション中にソースコードの正しい動作を妨げる何らかの欠陥または問題が見つかった場合、ソフトウェア開発者は、ソースコードに対して変更を行って欠陥または問題を解決し得る。いくつかのケースでは、変更は、デバッガが取り付けられずに行われ得る。それでも、別のシナリオは、デバッガが取り付けられるが、変更が行われるときにブレークモードでプログラムが中断されないことがあ

10

20

30

40

50

るときである。そのようなケースでは、ソースコードに対する変更が受け取られると、変更がタイプのソースコードにインプレースで適用され得る。言い換えれば、変更がタイプのソースコードに適用されているとき、ソースコードが上書きされている。変更が行われており、ソフトウェア開発者がアプリケーションを再開すると、タイプのランタイム表現およびその既存のインスタンスが更新され、変更が反映される。これは、プログラム全体を停止し、再コンパイルし、デバッグセッションを再始動することを必要とせず、ソフトウェア開発者がソースコードに対する変更を行うことを可能にする。しかしながら、タイプのランタイム表現およびそのインスタンスを更新することの固有の複雑さのために、可能にされる編集のセットは限定される。

【0002】

本明細書で開示される態様が作成されたのは、これらおよび他の一般的な考慮事項に関してである。さらに、比較的具体的な問題が論じられ得るが、実施例は、背景技術および本開示の別の箇所で特定される具体的な問題を解決することに限定されないことを理解されたい。

【発明の概要】

【0003】

本開示は、開発者がプログラムを停止、リロード、および再始動することなくプログラムを修正することを可能にすることによって開発者開発サイクルの内部ループを低減するホットリロードに関する。より具体的には、本開示は、タイプ（たとえば、クラス）が変化した場合にホットリロードする能力を教示する。この考えを実施するために、UIフレームワークと、NETランタイムが共に更新され得る。

【0004】

本開示の少なくとも1つの実施例によれば、ランタイムでのプログラムのソースコードに対する変更を実施するための方法が提供される。方法は、ランタイム環境でソースコードを実行するステップと、ランタイム環境でランタイムにソースコードに対して行われた変更を検出するステップと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定するステップと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたと判定したことに応答して、タイプの新しいバージョンを作成ステップことと、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うステップとを含み得る。

【0005】

本開示の少なくとも1つの実施例によれば、ランタイムにプログラムのソースコードに対して行われた変更を実施するためのコンピューティングデバイスが提供される。コンピューティングデバイスは、プロセッサと、複数の命令が記憶されたメモリとを含み得、複数の命令は、プロセッサによって実行されるとき、コンピューティングデバイスに、ランタイム環境でソースコードを実行することと、ランタイム環境でランタイムにソースコードに対して行われた変更を検出することと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたという判定に応答して、タイプの新しいバージョンを作成することと、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うこととをさせる。

【0006】

本開示の少なくとも1つの実施例によれば、ランタイムにプログラムのソースコードに対して行われた変更を実施するための命令を記憶する非一時的コンピュータ可読媒体が提供される。命令は、コンピューティングデバイスの1つまたは複数のプロセッサによって実行されるとき、コンピューティングデバイスに、ランタイム環境でソースコードを実行することと、ランタイム環境でランタイムにソースコードに対して行われた変更を検出することと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたという判定に応答して、タイプの新しいバージョンを作成することと、タイ

10

20

30

40

50

ブの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うこととをさせる。

【0007】

1つまたは複数の態様のうちの任意の他のものと組み合わせた1つまたは複数の上記の態様のいずれか。本明細書で説明される1つまたは複数の態様のいずれか。

【0008】

この概要は、以下の詳細な説明でさらに説明される概念の抜粋を簡略化した形で紹介するために与えられるものである。この概要は、特許請求される主題の主要な特徴または不可欠な特徴を特定するためのものではなく、特許請求される主題の範囲を限定するためのものでもない。実施例の追加の態様、特徴、および／または利点が、以下の説明で部分的に説明され、部分的には説明から明らかとなり、または本開示の実施によって知ることができる。

【0009】

以下の図を参照しながら、非限定的で非網羅的な実施例が説明される。

【図面の簡単な説明】

【0010】

【図1】本開示の実施例によるソフトウェア開発システムを対象とする詳細を示す図である。

【図2】本開示の実施例による、ランタイムにリロード可能タイプのソースコードに対して行われた変更を実施する方法を示す図である。

【図3】本開示の実施例による、ランタイムにリロード可能タイプのソースコードに対して行われた変更を実施する方法を示す図である。

【図4】本開示の態様が実施され得るコンピューティングデバイスの物理構成要素（たとえばハードウェア）を示すブロック図である。

【図5A】本開示の態様が実施され得るコンピューティングデバイスの第1の実施例を示す図である。

【図5B】本開示の態様が実施され得るコンピューティングデバイスの第2の実施例を示す図である。

【図6】本開示の実施例による、データを処理するためのシステムのアーキテクチャの少なくとも1つの態様を示す図である。

【発明を実施するための形態】

【0011】

以下の詳細な説明では、本明細書の一部を形成し、特定の態様または実施例が例として示される添付の図面に対して参照が行われる。本開示から逸脱することなく、これらの態様が組み合わされ得、他の態様が利用され得、構造的変更が行われ得る。態様は方法、システム、またはデバイスとして実施され得る。したがって、態様は、ハードウェア実施、完全にソフトウェアの実施、またはソフトウェア態様とハードウェア態様を組み合わせる実施の形態を取り得る。したがって、以下の詳細な説明は、限定的な意味で理解されるべきではなく、本開示の範囲は、添付の特許請求の範囲およびその均等物によって定義される。

【0012】

本開示の実施例によれば、ソフトウェア開発システムは、ユーザ（たとえば、開発者、プログラマ、設計者、および／またはコーダ）が、アプリケーションフレームワークを使用してプログラムを開発または修正することを可能にし、アプリケーションフレームワークは、アプリケーションソフトウェアの標準構造を実施するためのソフトウェアフレームワークを含む。本開示のソフトウェア開発システムは、プログラムが実行されている間に、プログラムのソースコードに対して行われた変更を実施するように構成される。具体的には、ソフトウェア開発システムは、プログラム全体を停止、リロード、および再始動することなくランタイムにソースコード内の特定のタイプをユーザが修正することを可能にする。

【0013】

そのように行うために、ソースコードまたはデータベースライブラリ内で特定のタイプがリロード可能とマークされ得る。いくつかの態様では、データベースライブラリがアプリケーションフレームワークに関連付けられ得る。リロード可能タイプのソースコードに対して変更が行われたとき、変更の結果、変更をインプレースで適用する（たとえば、タイプをインプレースで更新する）代わりに、リロード可能タイプの新しいバージョンを作成し得、リロード可能タイプの新しいバージョンがタイプの前のバージョンに取って代わり得る。そのように行うために、たとえば、アプリケーションフレームワークは、ソースコード内のリロード可能タイプの前のバージョンのランタイム表現および既存のインスタンスを、リロード可能タイプの新しいバージョンで置き換え得る。ソースコードに対して行われた変更を反映するタイプの新しいバージョンを作成することにより、ユーザは、インプレース更新の制限によって制約されることなく、ランタイム中にソースコードに対する変更を行うことができ得る。

10

【0014】

図1は、本開示の実施例による、ランタイムにユーザ110によってソースコードに対して行われた変更を実施するためのソフトウェア開発システム100を示す。そのように行うために、ソフトウェア開発システム100は、ユーザ110に関連付けられるコンピューティングデバイス120と、ネットワーク150を介してコンピューティングデバイス120に通信可能に結合される、サーバ140などの1つまたは複数のリモートデバイスを含む。ネットワーク150は、限定はしないが、ワイヤードもしくはワイヤレス・ローカル・エリア・ネットワーク（LAN）、ワイヤードもしくはワイヤレス広域ネットワーク（WAN）、および／またはインターネットを含む、任意の種類のコンピューティングネットワークを含み得る。

20

【0015】

コンピューティングデバイス120は、プロセッサ122、メモリ124、および通信インターフェース126を有するコンピューティングデバイス120上で実行される統合開発環境（IDE）130を含む。IDE130は、ユーザ110がコンピュータシステムでソフトウェアプログラム、プログラムのセット、ウェブサイト、ウェブアプリケーション、およびウェブサービスを設計、コーディング、コンパイル、テスト、実行、編集、デバッグ、または構築することを可能にし得る。ソフトウェアプログラムは、1つまたは複数のソースコード言語（たとえば、Visual Basic、Visual J#、C++、C#、J#、JavaScript、APL、COBOL、Pascal、Eiffel、Haskell、ML、Oberon、Perl、Python、Scheme、Smalltalkなど）で作成されたソースコードを含み得る。IDE130はネイティブコード開発環境を提供し得、または仮想マシン上で動作するマネージドコード環境を提供し得る。たとえば、IDE130は、.NETフレームワークを使用してマネージドコード開発環境を提供し得る。例示的实施形態では、IDE130は、たとえば、.NETフレームワークおよび／またはUIフレームワークと共に使用されるアプリケーションを開発するように構成され得る。ランタイムにリロード可能タイプのソースコードに対して行われた変更を実施するために、UIフレームワークと、.NETランタイムは共に更新され得る。

30

40

【0016】

たとえば、.NETフレームワークは、特定のタイプをリロード可能と指定する属性と、特定のタイプを更新通知の受信側として指定する属性で更新される。更新通知は、更新されたタイプについての情報を含む。属性は任意のタイプ上に配置され得るマーカであることを理解されたい。

【0017】

UIフレームワークはUI要素を表す特定のタイプを定義する。UIフレームワークがリロードすることのできる特定のタイプが、上記の属性でマークされる。UIフレームワークは、ランタイムから更新通知を受け取るように設計されるタイプを実施する。そのよ

50

うな通知の検索時に、UIフレームワークは、そのデータ構造内のすべての更新されたリロード可能タイプを、更新を含む、対応する新しいバージョンで置き換える。リロード可能タイプの使用パターンは、タイプをその新しいバージョンで置き換えることを可能にするようなものであることを理解されたい。言い換えれば、こうしたタイプを参照するユーザコードは、タイプの特定のバージョンを直接的に参照するのではなく、（たとえば、ベースタイプ、インターフェース、または動的ディスパッチを介して）間接的にこうしたタイプを参照する。

【0018】

そのように行うために、IDE 130は、ソースコードアナライザ132、リロード可能タイプ識別器134、およびフレームワーク・ライブラリ・アップデート136をさらに含む。ソースコードアナライザ132は、IDE 130でロードされたプログラムのソースコードを解析して、ランタイムにユーザ110によって行われた変更を検出するように構成される。例示的態様では、ユーザ110は、プログラムが実行されている間に、IDE 130を使用してソースコードに対する変更を行い得る。変更は、変更が行われたソースコードのタイプを示すメタデータに応じて、プログラム全体を停止、リロード／再コンパイル、および再始動することなくランタイムに実施され得る。上記で説明し、以下でさらに説明するように、あるタイプのソースコードはリロードすることができ、したがってプログラムが実行されている間に任意に更新され得る。そのようなリロード可能タイプのソースコードは、ユーザ110が、インプレースで更新されているタイプと比べて少ない制限で、ランタイムにプログラムを編集することを可能にする。リロード可能タイプ識別器134は、ソースコードに対して行われた変更が1つまたは複数のリロード可能タイプのソースコードに対して行われるかどうかを判定するように構成される。たとえば、ユーザ110は、開発プロセス中に、あるタイプにリロード可能と注釈を付けることによって1つまたは複数のリロード可能タイプを作成し得る。代替または追加として、フレームワークは、リロードすることのできる事前定義されたタイプのリロード可能タイプデータベースを含み得る。たとえば、アプリケーションフレームワークの開発者は、1つまたは複数のタイプを作成し、いくつかのタイプをリロード可能と識別し、識別したリロード可能タイプをリロード可能タイプデータベース内に記憶し得る。そのようなリロード可能タイプデータベースは、別のコンピューティングデバイス（たとえば、サーバ140）から受信され得る。サーバ140は1つまたは複数のクラウドサーバであり得る。

【0019】

フレームワーク・ライブラリ・アップデート136は、リロード可能タイプのソースコードの変更を示すメタデータを含む通知を別のデバイス（たとえば、サーバ140）に送信するように構成される。リロード可能タイプのソースコードに対して行われた変更は、リロード可能タイプの新しいバージョンを生成し、リロード可能タイプの新しいバージョンは、フレームワークに送られるメタデータ内に含まれる。いくつかの態様では、通知がサーバ140に送られ、フレームワークが、サーバ140のフレームワーク・ライブラリ・アップデート142を介して更新され得る。リロード可能タイプの新しいバージョンがフレームワークライブラリ（たとえば、リロード可能タイプデータベース128および／またはフレームワークライブラリ144）内に記憶される。それに応答して、フレームワークは、それ自体のランタイム表現を有する新しいタイプ（たとえば、タイプの新しいバージョン）を作成する。UIフレームワークにリロード可能タイプの更新が通知されたとき、UIフレームワークは、タイプの古いバージョンに対する、UIフレームワークが有するすべての参照を、新しいバージョンで置き換える。UIがリフレッシュされるとき、UIフレームワークは、どのタイプが特定のUI要素を実施するかを識別し、そのタイプをUIフレームワークのデータ構造からフェッチする。これはタイプの最新バージョンとなる。その後で、UIフレームワークは、要素のUIをレンダリングするタイプ内のコードを実行する。これは、プログラムが実行されている間に変更を直ちに実施することを可能にする。

【0020】

リロード可能タイプを使用し得るフレームワークの一例は `Razor` である。`Razor` は、ウェブアプリケーションおよび／またはウェブページを作成するために使用されるフレームワークである。`Razor` は、プログラマが `HTML` 構築ワークフローを使用することを可能にする。以下で示される実施例では、`Microsoft.AspNetCore.Mvc.RazorPages.Page` が、ウェブページを表すユーザ定義のタイプのベースタイプである。このタイプは通常、`Razor` 言語でページを記述する。`razor` ファイルから生成される。これは、以下のように見える `C#` コードに自動的に変換される。

【0021】

```
public class MyPage:Microsoft.AspNetCore.Mvc.RazorPages.Page
{
    public async override Task ExecuteAsync()
    {
        // ... ページのHTMLコンテンツを構築するコード ...
    }
}
```

10

ユーザ110が、`razor` ファイルを更新する（たとえば、ページのタイトルを変更する）とき、それに応じて上記のタイプのソースが自動的に更新される。`Razor` コメントは、ページがブラウザに送られる前にサーバ140上で処理される。ウェブアプリケーションが実行されている間にこうした更新が直ちに効力を生ずることを可能にするために、`Razor` フレームワークは、ベースタイプ `Microsoft.AspNetCore.Mvc.RazorPages.Page` をリロード可能とマークし、その結果、ページのすべてのサブタイプがリロード可能となる。これは、アプリケーションを終了し、再コンパイルし、再び実行する必要をなくし、インプレースで行われ得る更新に関するどんな制約も回避する。

20

【0022】

リロード可能タイプの別の例は `Model-View-Update (MVU)` パターンである。`Comet UI` フレームワークが `MVU` パターンを使用して構築される。`Comet UI` フレームワークは、以下のようにアプリケーション `UI` ビューが導出される `View` クラスを提供する。

30

【0023】

```
public class My View:View
{
    [Body]
    View body()=> new Text("Hello World");
}
```

`Comet` フレームワークは、`View` タイプを「リロード可能」とマークして、アプリケーションが実行されている間にユーザ110がビューを更新することを可能にする。

【0024】

次に図2および3を参照して、本開示の実施例による、プログラムが実行されている間にプログラムのソースコードに対して行われた変更を実施する方法200が提供される。方法200のステップについての全体的な順序が図2および3に示されている。一般に、方法200は204で開始し、254で終了する。方法200は、より多くの、もしくは少ないステップを含み得、図2および3に示されるのとは異なるようにステップの順序を構成し得る。例示的態様では、方法200は、ユーザ110のコンピューティングデバイス（たとえば、ユーザデバイス120）によって行われる。具体的には、いくつかの態様では、方法200は、ユーザデバイス120上で実行される統合開発環境（`IDE`）（たとえば、130）によって行われ得る。たとえば、コンピューティングデバイス120は、限定はしないが、サーバ140と通信することのできるコンピュータ、ノートブック、ラップトップ、モバイルデバイス、スマートフォン、タブレット、ウェアラブル、または

40

50

任意の他の適切なコンピューティングデバイスであり得る。たとえば、サーバ 140 は、コンピューティングデバイス 120 と通信することのできる任意の適切なコンピューティングデバイスであり得る。方法 200 は、コンピュータシステムによって実行され、コンピュータ可読媒体上で符号化または記憶されるコンピュータ実行可能命令として実行され得る。さらに、方法 200 は、プロセッサ、特定用途向け集積回路 (ASIC)、フィールドプログラマブルゲートアレイ (FPGA)、システムオンチップ (SOC)、または他のハードウェアデバイスに関連付けられるゲートまたは回路によって行われ得る。以下では、方法 200 は、図 1 と共に説明されるシステム、構成要素、モジュール、ソフトウェア、データ構造、ユーザインターフェースなどを参照して説明される。

【0025】

方法 200 は 202 で開始し、フローは 206 に進み得る。206 では、コンピューティングデバイス 120 が、統合開発環境 (IDE) (たとえば、124) でプログラムのソースコードを取得または構築する。いくつかの態様では、ソースコードはソフトウェアフレームワーク (たとえば、.NET フレームワーク) 内で構築され得る。たとえば、ソースコードはユーザ 110 によって提供され得る。代替として、ソースコードは、サーバ 140 から受信または取得され得る。ソースコードが取得されると、ソースコードが IDE 124 内にロードされる。たとえば、IDE 124 は、コンピューティングデバイス 120 上で実行される Microsoft (登録商標) Visual Studio (登録商標)、Eclipse、Microsoft (登録商標) Office、または任意の他の IDE であり得る。

【0026】

その後で、210 では、IDE 124 によって提供されるランタイム環境でソースコードが実行される。いくつかの態様では、コンピューティングデバイス 120 は、ソースコードを解析して、ランタイム中にリロード可能と識別される 1 つまたは複数のタイプを識別し得る。たとえば、ユーザは、1 つまたは複数のタイプを作成し、そうしたタイプに、開発プロセス中にリロード可能と注釈を付け得る。代替または追加として、ソフトウェアフレームワークは、ランタイムにリロードすることのできる、事前定義されたタイプのリロード可能タイプライブラリまたはデータベースを含み得る。リロード可能タイプデータベースは、別のコンピューティングデバイス (たとえば、サーバ 140) から受信され得る。代替または追加として、ソフトウェアフレームワークの開発者は、1 つまたは複数のタイプを作成し、あるタイプをリロード可能であると識別し、識別したリロード可能タイプをリロード可能タイプデータベース内に記憶し得る。218 では、プログラムがランタイム環境内で実行されている間に、コンピューティングデバイス 120 は、ユーザ 110 によって行われたソースコードに対する 1 つまたは複数の変更を検出する。1 つまたは複数の変更が検出されたとき、コンピューティングデバイス 120 は、動作 222 に記載されているように、1 つまたは複数の変更がリロード可能タイプのソースコードに対して行われたかどうかを判定する。前述のように、ソフトウェアフレームワークに関連付けられるリロード可能タイプデータベース 128 は、ランタイムにリロードすることのできる、事前定義されたタイプを含み得る。そのような態様では、コンピューティングデバイス 120 は、リロード可能タイプデータベース 128 に部分的に基づいて、1 つまたは複数の変更がリロード可能タイプのソースコードに対して行われたかどうかを判定する。追加または代替として、ランタイムの前および／または間に、ソースコード内の特定のタイプが識別され、ユーザ 110 によってリロード可能タイプのソースコードと注釈が付けられ得る。追加または代替として、ユーザ 110 は新しいタイプを作成し、新しいタイプにリロード可能と注釈を付け得る。その後で、方法 200 は、図 2 および 3 の英数文字 A で示されるように、図 3 の動作 226 に進む。

【0027】

226 では、変更がリロード可能タイプのソースコード内にないとコンピューティングデバイス 120 が判定した場合、方法 200 は動作 230 に進み、リロード不能なタイプのソースコードに対して行われた変更をインプレースで適用する。言い換えれば、リロー

10

20

30

40

50

ド不能なタイプのランタイム表現およびそのすべての既存のインスタンスが更新され、変更が反映される。しかしながら、前述のように、タイプのランタイム表現およびそのインスタンスを更新することの固有の複雑さのために、可能にされるインプレース編集のセットは限定される。次いで、方法 200 は、前方に動作 250 までスキップし、更新後のソースコードと共にランタイム環境内のプログラムの実行を続行する。

【0028】

動作 226 まで戻ると、リロード可能タイプのソースコード内で変更が行われるとコンピューティングデバイス 120 が判定した場合、方法 200 は動作 234 に進む。234 では、コンピューティングデバイス 120 は、変更に基づいて新しいタイプ（たとえば、リロード可能タイプの新しいバージョン）を作成する。

【0029】

その後で、238 では、コンピューティングデバイス 120 は、ソフトウェアフレームワークによってリロードする必要のあるタイプの新しいバージョンについての情報（たとえば、メタデータ）をソフトウェアフレームワークに通知する。通知を受け取ったことに応答して、242 では、ソフトウェアフレームワークは、有意な状態（たとえば、メンバ変数）がある場合にそれを、リロード可能タイプの前のインスタンスから、リロード可能タイプの新しいバージョンに移す。

【0030】

246 では、コンピューティングデバイス 120 は、ソースコード内のリロード可能タイプの前のバージョンのランタイム提示および既存のインスタンスを、リロード可能タイプの新しいバージョンに置き換える。

【0031】

その後で、250 では、コンピューティングデバイス 120 は、プログラムが実行されている間に変更を実施した後、ランタイム環境内のプログラムの実行を続行する。これは、プログラム全体を停止、リロード、および再始動することなくランタイムにソースコードに対する変更をユーザ 110 が行うことを可能にする。

【0032】

図 4～6 および関連する説明は、本開示の態様が実施され得る様々な動作環境の議論を提供する。しかしながら、図 4～6 に関連して図示され、論じられるデバイスおよびシステムは、例示および図示のためのものであり、本明細書で説明される本開示の態様を実施するために利用され得る莫大な数のコンピューティングデバイス構成の限定ではない。

【0033】

図 4 は、本開示の態様が実施され得るコンピューティングデバイス 400 の物理構成要素（たとえばハードウェア）を示すブロック図である。以下で説明されるコンピューティングデバイス構成要素は、前述のコンピューティングデバイスに適したものであり得る。たとえば、コンピューティングデバイス 400 は図 1 のコンピューティングデバイス 120 を表し得る。基本的構成では、コンピューティングデバイス 400 は、少なくとも 1 つの処理装置 402 およびシステムメモリ 404 を含み得る。コンピューティングデバイスの構成およびタイプに応じて、システムメモリ 404 は、限定はしないが、揮発性ストレージ（たとえばランダムアクセスメモリ）、不揮発性ストレージ（たとえば読取り専用メモリ）、フラッシュメモリ、またはそのようなメモリの任意の組合せを含み得る。

【0034】

システムメモリ 404 は、オペレーティングシステム 405 と、そのように本明細書で開示される様々な態様を行うのに適した 1 つまたは複数のプログラムモジュール 406 とを含み得る。オペレーティングシステム 405 は、たとえば、コンピューティングデバイス 400 の動作を制御するのに適しているものであり得る。さらに、本開示の態様は、グラフィックスライブラリ、他のオペレーティングシステム、または任意の他のアプリケーションプログラムと共に実施され得、何らかの特定のアプリケーションまたはシステムに限定されない。この基本的構成が、図 4 に、破線 408 内のそうした構成要素によって示されている。コンピューティングデバイス 400 は追加の特徴または機能を有し得る。た

10

20

30

40

50

たとえば、コンピューティングデバイス 400 はまた、たとえば、磁気ディスク、光ディスク、テープなどの追加のデータ記憶デバイス（取外し可能および／または取外し不能）をも含み得る。そのような追加のストレージが、図 4 に、取外し可能記憶デバイス 409 および取外し不能記憶デバイス 410 によって示されている。

【0035】

前述のように、いくつかのプログラムモジュールおよびデータファイルが、システムメモリ 404 内に記憶され得る。少なくとも 1 つの処理装置 402 上で実行されている間、プログラムモジュール 406 は、限定はしないが、本明細書で説明される 1 つまたは複数の状態を含むプロセスを行い得る。アプリケーション 420 は、図 1 に関してより詳細に説明されるように、ソースコードアナライザ 423、リロード可能タイプ識別器 424、フレームワーク・ライブラリ・アップデート 425 を含む。本開示の状態に従って使用され得る他のプログラムモジュールは、電子メールおよび連絡先アプリケーション、ワード・プロセッシング・アプリケーション、スプレッドシートアプリケーション、データベースアプリケーション、スライド・プレゼンテーション・アプリケーション、ドローイングまたはコンピュータ支援アプリケーションプログラムなど、ならびに／あるいは本明細書で説明されるシステムによってサポートされる 1 つまたは複数の構成要素を含み得る。

10

【0036】

さらに、本開示の状態は、ディスクリット電子素子を備える電気回路、論理ゲートを含むパッケージ化または集積された電子チップ、マイクロプロセッサを使用する回路で、あるいは電子素子またはマイクロプロセッサを含むシングルチップ上で実施され得る。たとえば、本開示の状態は、システムオンチップ（SOC）を介して実施され得、図 4 に示される構成要素のうちのそれぞれまたは多くが単一の集積回路上で集積され得る。そのような SOC デバイスは、1 つまたは複数の処理装置、グラフィックスユニット、通信ユニット、システム仮想化ユニット、および様々なアプリケーション機能を含み得、そのすべてが、チップ基板上で単一の集積回路として集積される（または「焼き付けられる」）。SOC を介して動作するとき、プロトコルを切り換えるクライアントの能力に関する、本明細書で説明される機能が、単一の集積回路（チップ）上でコンピューティングデバイス 400 の他の構成要素と共に集積された特定用途向けロジックを介して動作し得る。本開示の状態はまた、限定はしないが機械、光学、流体、および量子技術を含む、たとえば AND、OR、NOT などの論理演算を行うことのできる他の技術を使用して実施され得る。

20

30

【0037】

さらに、本開示の状態は、汎用コンピュータ、あるいは任意の他の回路またはシステム内で実施され得る。

【0038】

コンピューティングデバイス 400 はまた、キーボード、マウス、ペン、サウンドまたは音声入力デバイス、タッチまたはスワイプ入力デバイスなどの 1 つまたは複数の入力デバイス 412 をも有し得る。ディスプレイ、スピーカ、プリンタなどの出力デバイス 414 A も含まれ得る。仮想ディスプレイに対応する出力 414 B も含まれ得る。前述のデバイスは例であり、他のものも使用され得る。コンピューティングデバイス 400 は、他のコンピューティングデバイス 450 との通信を可能にする 1 つまたは複数の通信接続 416 を含み得る。適切な通信接続 416 の例は、限定はしないが、無線周波数（RF）送信機、受信機、および／またはトランシーバ回路、ユニバーサルシリアルバス（USB）、パラレルおよび／またはシリアルポートを含む。

40

【0039】

本明細書では、コンピュータ可読媒体という用語はコンピュータ記憶媒体を含み得る。コンピュータ記憶媒体は、コンピュータ可読命令、データ構造、プログラムモジュールなどの情報の記憶のための任意の方法または技術で実施された揮発性および不揮発性の取外し可能および取外し不能媒体を含み得る。システムメモリ 404、取外し可能記憶デバイス 409、および取外し不能記憶デバイス 410 はすべてコンピュータ記憶媒体の例（たとえば、メモリストレージ）である。コンピュータ記憶媒体は、RAM、ROM、電気消

50

去可能読取り専用メモリ（EEPROM）、フラッシュメモリ、または他のメモリ技術、CD-ROM、デジタル・バーサタイル・ディスク（DVD）、または他の光ストレージ、磁気カセット、磁気テープ、磁気ディスクストレージ、または他の磁気記憶デバイス、あるいは情報を記憶するために使用され得、コンピューティングデバイス400によってアクセスされ得る任意の他の製造品を含み得る。任意のそのようなコンピュータ記憶媒体はコンピューティングデバイス400の部分であり得る。コンピュータ記憶媒体は、搬送波あるいは他の伝播または被変調データ信号を含まない。

【0040】

通信媒体は、搬送波や他の移送機構などの被変調データ信号内のコンピュータ可読命令、データ構造、プログラムモジュール、または他のデータで具現化され得、任意の情報伝達媒体を含む。「被変調データ信号」という用語は、信号内に情報を符号化するような方式で1つまたは複数の特性が設定または変更された信号を記述し得る。限定ではなく例として、通信媒体は、ワイヤードネットワークやダイレクトワイヤード接続などのワイヤード媒体と、音響、無線周波数（RF）、赤外線、および他のワイヤレス媒体などのワイヤレス媒体とを含み得る。

【0041】

図5Aおよび5Bは、本開示の態様が実施され得るコンピューティングデバイスまたはモバイル・コンピューティング・デバイス500、たとえば、携帯電話、スマートフォン、ウェアラブルコンピュータ（スマートウォッチなど）、タブレットコンピュータ、ラップトップコンピュータなどを示す。図5Aを参照すると、態様を実施するためのモバイル・コンピューティング・デバイス500の一態様が示されている。基本的構成では、モバイル・コンピューティング・デバイス500は、入力要素と出力要素を共に含むハンドヘルドコンピュータである。モバイル・コンピューティング・デバイス500は通常、ディスプレイ505と、ユーザがモバイル・コンピューティング・デバイス500内に情報を入力することを可能にする1つまたは複数の入力ボタン509／510とを含む。モバイル・コンピューティング・デバイス500のディスプレイ505は、入力デバイス（たとえば、タッチスクリーンディスプレイ）としても機能し得る。含まれる場合、任意選択のサイド入力要素515が別のユーザ入力を可能にする。サイド入力要素515は、回転スイッチ、ボタン、または任意の他のタイプの手動入力要素であり得る。代替の態様では、モバイル・コンピューティング・デバイス500は、より多くの要素または少ない要素を組み込み得る。たとえば、いくつかの態様では、ディスプレイ505はタッチスクリーンではないことがある。さらに別の代替の態様では、モバイル・コンピューティング・デバイス500は、セルラフォンなどのポータブル・フォン・システムである。モバイル・コンピューティング・デバイス500はまた、任意選択のキーパッド535をも含み得る。任意選択のキーパッド535は、物理キーパッド、またはタッチスクリーンディスプレイ上に生成された「ソフト」キーパッドであり得る。様々な態様では、出力要素は、グラフィカル・ユーザ・インターフェース（GUI）を示すためのディスプレイ505、視覚インジケータ531（たとえば、発光ダイオード）、および／またはオーディオ変換器525（たとえば、スピーカ）を含む。いくつかの態様では、モバイル・コンピューティング・デバイス500は、触覚フィードバックをユーザに提供するための振動変換器を組み込む。さらに別の態様では、モバイル・コンピューティング・デバイス500は、外部ソースに信号を送り、または外部ソースから信号を受信するための、オーディオ入力（たとえば、マイクロフォンジャック）、オーディオ出力（たとえば、ヘッドフォンジャック）、ビデオ出力（たとえば、HDMIポート）などの入力および／または出力ポート530を組み込む。

【0042】

図5Bは、コンピューティングデバイス、サーバ、またはモバイル・コンピューティング・デバイスの一態様のアーキテクチャを示すブロック図である。すなわち、モバイル・コンピューティング・デバイス500は、いくつかの態様を実施するためにシステム（902）（たとえば、アーキテクチャ）を組み込み得る。システム502は、1つまたは複

10

20

30

40

50

数のアプリケーション（たとえば、ブラウザ、eメール、カレンダー、連絡先マネージャ、メッセージングクライアント、ゲーム、メディアクライアント／プレーヤ）を実行することのできる「スマートフォン」として実施され得る。いくつかの態様では、システム502は、一体型携帯情報端末（PDA）およびワイヤレスフォンなどのコンピューティングデバイスとして一体化される。

【0043】

1つまたは複数のアプリケーションプログラム566がメモリ562内にロードされ、オペレーティングシステム564上で、またはオペレーティングシステム564に関連して実行され得る。アプリケーションプログラムの例は、フォンダイヤラプログラム、eメールプログラム、個人情報管理（PIM）プログラム、ワード・プロセッシング・プログラム、スプレッドシートプログラム、インターネット・ブラウザ・プログラム、メッセージングプログラム、ならびに／あるいは本明細書で説明されるシステムによってサポートされる1つまたは複数の構成要素を含む。システム502はまた、メモリ562内の不揮発性記憶エリア568をも含む。不揮発性記憶エリア568は、システム502が電源遮断される場合に失われるべきではない永続的情報を記憶するために使用され得る。アプリケーションプログラム566は、eメールアプリケーションによって使用されるeメールや他のメッセージなどの、不揮発性記憶エリア568内の情報を使用し、不揮発性記憶エリア568内に情報を記憶し得る。同期アプリケーション（図示せず）もシステム502上に常駐し、ホストコンピュータ上に常駐する、対応する同期アプリケーションと対話して、不揮発性記憶エリア568内に記憶された情報を、ホストコンピュータに記憶された、対応する情報と同期されるように保つようにプログラムされる。理解すべきであるが、他のアプリケーションがメモリ562内にロードされ、本明細書で説明されるモバイル・コンピューティング・デバイス500（たとえば、ソースコードアナライザ423、リロード可能タイプ識別器424、フレームワーク・ライブラリ・アップデート425など）上で実行され得る。システム502は電源570を有し、電源570は1つまたは複数の電池として実施され得る。電源570は、電池を補足または充電するACアダプタや電源付きドッキングクレードルなどの外部電源をさらに含み得る。

【0044】

システム502はまた、無線周波数通信を送信および受信する機能を行う無線インターフェース層572をも含み得る。無線インターフェース層572は、通信キャリアまたはサービスプロバイダを介する、システム502と「外部世界」との間のワイヤレス接続性を容易にする。無線インターフェース層572との間の伝送は、オペレーティングシステム564の制御下で行われる。言い換えれば、無線インターフェース層572によって受信された通信が、オペレーティングシステム564を介してアプリケーションプログラム566に流布され得、逆も同様である。

【0045】

視覚インジケータ520が、視覚的通知を提供するために使用され得、および／またはオーディオインターフェース574が、オーディオ変換器525を介して可聴通知を生成するために使用され得る。図示される構成では、視覚インジケータ520は発光ダイオード（LED）であり、オーディオ変換器525はスピーカである。これらのデバイスは、電源570に直接的に結合され得、その結果、プロセッサ560／961および他の構成要素が電池電力を保持するためにシャットダウンされ得るとしても、これらのデバイスは、活動化されるとき、通知機構によって指示される持続時間にわたってオンのままとなる。LEDは、デバイスの電源オンステータスを示すために、ユーザが行動を起こすまで無期限にオンのままとなるようにプログラムされ得る。オーディオインターフェース574は、ユーザに可聴信号を提供し、ユーザから可聴信号を受け取るために使用される。たとえば、オーディオ変換器525に結合されることに加えて、オーディオインターフェース574は、電話会話を容易にするなどのために、可聴入力を受け取るためにマイクロフォンにも結合され得る。本開示の態様によれば、マイクロフォンはまた、以下で説明されるように、通知の制御を容易にするためにオーディオセンサとしても働き得る。システム5

02は、静止画像、ビデオストリームなどを記録するためにオンボードカメラの操作を可能にするビデオインターフェース576をさらに含む得る。

【0046】

システム502を実施するモバイル・コンピューティング・デバイス500は、追加の特徴または機能を有し得る。たとえば、モバイル・コンピューティング・デバイス500はまた、磁気ディスク、光ディスク、テープなどの追加のデータ記憶デバイス（取外し可能および／または取外し不能）をも含む得る。そのような追加のストレージが、図5Bに、不揮発性記憶エリア568によって示されている。

【0047】

モバイル・コンピューティング・デバイス500によって生成または取り込まれ、システム502を介して記憶されたデータ／情報は、前述のようにモバイル・コンピューティング・デバイス500上にローカルに記憶され得、またはデータは、無線インターフェース層572を介して、またはモバイル・コンピューティング・デバイス500と、モバイル・コンピューティング・デバイス500に関連付けられる別々のコンピューティングデバイス、たとえば、インターネットなどの分散型コンピューティングネットワーク内のサーバコンピュータとの間のワイヤード接続を介してデバイスによってアクセスされ得る任意の数の記憶媒体上に記憶され得る。理解すべきであるが、そのようなデータ／情報は、モバイル・コンピューティング・デバイス500を介し、無線インターフェース層572を介して、または分散型コンピューティングネットワークを介してアクセスされ得る。同様に、そのようなデータ／情報は、電子メールおよび共同データ／情報共有システムを含む周知のデータ／情報転送および記憶手段に従う記憶および使用のために、コンピューティングデバイス間で容易に転送され得る。

【0048】

図6は、前述のように、パーソナルコンピュータ604、タブレット・コンピューティング・デバイス606、モバイル・コンピューティング・デバイス608などのリモートソースからコンピューティングシステムで受信されたデータを処理するためのシステムのアーキテクチャの一態様を示す。サーバデバイス602で表示されるコンテンツは、様々な通信チャネルまたは他の記憶タイプ内に記憶され得る。たとえば、コンピューティングデバイス604、606、608は図1のコンピューティングデバイス120を表し得、サーバデバイス602は図1のサーバ140を表し得る。

【0049】

いくつかの態様では、ソースコードアナライザ623、信頼できるタイプ識別器624、およびフレームワーク・ライブラリ・アップデータ625のうちの1つまたは複数が、サーバデバイス602によって利用され得る。サーバデバイス602は、ネットワーク612を通じて、パーソナルコンピュータ604、タブレット・コンピューティング・デバイス606、および／またはモバイル・コンピューティング・デバイス608（たとえば、スマートフォン）などのクライアントコンピューティングデバイスとの間でデータを提供し得る。例として、前述のコンピュータシステムは、パーソナルコンピュータ604、タブレット・コンピューティング・デバイス606、および／またはモバイル・コンピューティング・デバイス608（たとえば、スマートフォン）で具現化され得る。コンピューティングデバイスのこうした態様のいずれかは、グラフィック発信側システムで前処理し、または受信側コンピューティングシステムで後処理するために使用可能なグラフィカルデータを受信することに加えて、ストア616からコンテンツを取得し得る。コンテンツストアはリロード可能タイプデータ618を含む得る。

【0050】

図6は、本明細書で開示される1つまたは複数の態様を実行し得る例示的モバイル・コンピューティング・デバイス608を示す。さらに、本明細書で説明される態様および機能は、分散型システム（たとえば、クラウドベースのコンピューティングシステム）を介して動作し得、分散型システムでは、アプリケーション機能、メモリ、データ記憶および検索、ならびに様々な処理機能が、インターネットやイントラネットなどの分散型コンピ

10

20

30

40

50

ューティングネットワークを介して互いにリモートに操作され得る。ユーザインターフェースおよび様々なタイプの情報が、オンボード・コンピューティング・デバイス・ディスプレイを介して、または1つまたは複数のコンピューティングデバイスに関連付けられるリモート・ディスプレイ・ユニットを介して表示され得る。たとえば、ユーザインターフェースおよび様々なタイプの情報が、ユーザインターフェースおよび様々なタイプの情報が投影される壁面上に表示され、壁面と対話され得る。本発明の態様が実施され得る多数のコンピューティングシステムとの対話は、キーストローク入力、タッチスクリーン入力、音声または他のオーディオ入力、ジェスチャ入力などを含み、ジェスチャ入力では、関連するコンピューティングデバイスが、コンピューティングデバイスの機能を制御するためのユーザジェスチャを取り込んで解釈するための検出（たとえば、カメラ）機能を備える。

10

【0051】

「少なくとも1つ」、「1つまたは複数」、「または」、および「および／または」は、演算の際に接続的であると共に離散的である非限定的表現である。たとえば、「A、B、およびCのうちの少なくとも1つ」、「A、B、またはCのうちの少なくとも1つ」、「A、B、およびCのうちの1つまたは複数」、「A、B、またはCのうちの1つまたは複数」、「A、B、および／またはC」、「およびA、B、またはC」という表現のそれぞれは、Aのみ、Bのみ、Cのみ、AとB、AとC、BとC、またはA、B、およびCを意味する。

【0052】

エンティティ（“a” or “an” entity）という用語は、そのエンティティの1つまたは複数を指す。したがって、「a」（または「an」）、「1つまたは複数」、および「少なくとも1つ」は本明細書では互換的に使用され得る。「備える（comprising）」、「含む（including）」、および「有する（having）」は互換的に使用され得る。

20

【0053】

「自動」という用語およびその変形は、本明細書では、プロセスまたは動作が行われるとき、通常は連続的または半連続的であり、本質的な人間の入力なしに行われる任意のプロセスまたは動作を指す。しかしながら、プロセスまたは動作を行うことが本質的な、または本質的ではない人間の入力を使用するとしても、プロセスまたは動作を行う前に入力を受け取られる場合、プロセスまたは動作は自動であり得る。人間の入力が、どのようにプロセスまたは動作が行われるかに影響を及ぼす場合、そのような入力は本質的であると見なされる。プロセスまたは動作を行うことに同意する人間の入力は、「本質的」とは見なされない。

30

【0054】

本明細書で論じられるステップ、機能、および動作のいずれかは、継続的かつ自動的に行われ得る。

【0055】

本開示の例示的システムおよび方法が、コンピューティングデバイスに関連して説明された。しかしながら、本開示を不必要に曖昧にするのを避けるために、前述の説明はいくつかの周知の構造およびデバイスを省略した。この省略を限定と解釈すべきではない。本開示の理解を与えるために特定の詳細が説明される。しかしながら、本明細書で説明される特定の詳細以外に様々な方式で本開示が実施され得ることを理解されたい。

40

【0056】

さらに、本明細書で示される例示的態様は、共に配置されたシステムの様々な構成要素を示すが、システムのいくつかの構成要素は、LANおよび／またはインターネットなどの分散型ネットワークの離れた部分に、または専用システム内に、リモートに配置され得る。したがって、システムの構成要素が、サーバ、通信デバイスなどの1つまたは複数のデバイスとして組み合わせられ、あるいはアナログおよび／またはデジタル電気通信ネットワーク、パケット交換ネットワーク、回線交換ネットワークなどの分散型ネットワークの

50

特定のノード上に共に配置され得ることを理解されたい。前述の説明から、計算効率のために、システムの構成要素が、システムの動作に影響を及ぼすことなく、構成要素の分散型ネットワーク内の任意の場所に配置され得ることを理解されよう。

【0057】

さらに、要素を接続する様々なリンクが、ワイヤードもしくはワイヤレスリンク、またはそれらの任意の組合せ、あるいは接続された要素との間でデータを供給および／または通信することのできる任意の他の周知の要素または後で開発される要素であり得ることを理解されたい。こうしたワイヤードまたはワイヤレスリンクはセキュアリンクでもあり得、暗号化情報を通信することができ得る。たとえばリンクとして使用される伝送媒体は、同軸ケーブル、銅線、および光ファイバを含む電気信号のための任意の適切なキャリアであり得、電波および赤外線データ通信中に生成されるような音響波または光波の形態を取り得る。

【0058】

イベントの特定のシーケンスに関連してフローチャートが論じられ、図示されたが、開示される構成および態様の動作に本質的に影響を及ぼすことなく、このシーケンスに対する変更、追加、および省略が行われ得ることを理解されたい。

【0059】

本開示のいくつかの変形形態および修正形態が使用され得る。本開示のいくつかの特徴を、他の特徴を提供することなく提供することが可能となる。

【0060】

さらに別の構成では、本開示のシステムおよび方法は、専用コンピュータ、プログラムされたマイクロプロセッサまたはマイクロコントローラおよび周辺集積回路素子、ASICまたは他の集積回路、デジタル信号プロセッサ、ディスクリット素子回路などのハードワイヤード電子または論理回路、PLD、PLA、FPGA、PALなどのプログラマブル論理デバイスまたはゲートアレイ、専用コンピュータ、任意の同程度の手段などと共に実施され得る。一般には、本明細書で示される方法を実施することのできる任意のデバイスまたは手段が、本開示の様々な態様を実施するために使用され得る。本開示のために使用され得る例示的ハードウェアは、コンピュータ、ハンドヘルドデバイス、電話（たとえば、セルラ、インターネット対応、デジタル、アナログ、ハイブリッド、およびその他）、および当技術分野で周知の他のハードウェアを含む。こうしたデバイスのいくつかは、プロセッサ（たとえば、単一または複数のマイクロプロセッサ）、メモリ、不揮発性ストレージ、入力デバイス、および出力デバイスを含む。さらに、限定はしないが、分散処理またはコンポーネント／オブジェクト分散処理、並列処理、または仮想マシン処理を含む代替ソフトウェア実施も、本明細書で説明される方法を実施するように構築され得る。

【0061】

さらに別の構成では、開示される方法は、様々なコンピュータまたはワークステーションプラットフォーム上で使用され得るポータブルソースコードを提供するオブジェクトまたはオブジェクト指向ソフトウェア開発環境を使用するソフトウェアと共に容易に実施され得る。代替として、開示されるシステムは、標準論理回路またはVLSI設計を使用して、ハードウェアで部分的または完全に実施され得る。本開示によるシステムを実施するためにソフトウェアが使用されるか、それともハードウェアが使用されるかは、システムの速度および／または効率要件、特定の機能、ならびに使用される特定のソフトウェアまたはハードウェアシステムまたはマイクロプロセッサまたはマイクロコンピュータシステムに依存する。

【0062】

さらに別の構成では、開示される方法は、部分的には、記憶媒体上に記憶され、コントローラおよびメモリと協働するプログラムされた汎用コンピュータ、専用コンピュータ、マイクロプロセッサなどの上で実行され得るソフトウェアで実施され得る。こうした例では、本開示のシステムおよび方法は、アプレット、JAVA（登録商標）、CGIスクリプトなどのパーソナルコンピュータ上に組み込まれたプログラムとして、サーバまたはコ

ンピュータワークステーション上に常駐するリソースとして、専用測定システム、システム構成要素内に組み込まれたルーチンなどとして実施され得る。システムはまた、ソフトウェアおよび／またはハードウェアシステム内にシステムおよび／または方法を物理的に組み込むことによって実施され得る。

【0063】

本開示は、規格およびプロトコルが記述される場合、規格およびプロトコルに限定されない。本明細書で述べられていない他の類似の規格およびプロトコルが存在し、本開示に含まれる。さらに、本明細書で述べられる規格およびプロトコル、ならびに本明細書で述べられていない他の類似の規格およびプロトコルは、本質的に同一の機能を有する、より高速な、またはより効果的な均等物に周期的に取って代わられる。同一の機能を有するそのような代替規格およびプロトコルは、本開示内に含まれる均等物と見なされる。

10

【0064】

本開示の少なくとも1つの例によれば、ランタイムにプログラムのソースコードに対して行われた変更を実施するための方法が提供される。方法は、ランタイム環境でソースコードを実行することと、ランタイム環境でランタイムにソースコードに対して行われた変更を検出することと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することと、リロード可能と識別されたソースコード内のタイプに対して変更が行われたと判定したことに応答して、タイプの新しいバージョンを作成することと、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うこととを含み得る。

20

【0065】

上記の方法の少なくとも1つの態様によれば、方法は、ランタイム環境でソースコードを実行することが、ソースコードを取得することと、統合開発環境（IDE）内にソースコードをロードすること、IDEによって配置されたランタイム環境でソースコードを実行することとを含むことを含み得る。

【0066】

上記の方法の少なくとも1つの態様によれば、方法は、リロード可能と識別されたソースコード内のタイプに対して変更が行われなかったと判定したことに応答して、タイプをインプレースで更新することをさらに含み得る。

【0067】

上記の方法の少なくとも1つの態様によれば、方法は、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うことが、ソースコード内のタイプの前のバージョンのランタイム表現および1つまたは複数の既存のインスタンスを、タイプの新しいバージョンに置き換えることをさらに含むことを含み得る。

30

【0068】

上記の方法の少なくとも1つの態様によれば、方法は、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うことが、ソースコードを構築するために使用されたフレームワークに、タイプの新しいバージョンが作成されたことを通知することと、ソースコード内のタイプの前のバージョンの1つまたは複数の既存のインスタンスから新しいバージョンに状態を移すこととをさらに含むことを含み得る。

40

【0069】

上記の方法の少なくとも1つの態様によれば、方法は、ランタイム環境内の変更を伴う更新後のソースコードの実行を続行することをさらに含み得る。

【0070】

上記の方法の少なくとも1つの態様によれば、方法は、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することが、ソースコードを構築するために使用されたフレームワークに関連付けられるフレームワークライブラリに基づいて、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することをさらに含むことを含み得、フレームワークライブラリは、リロード可能とマークされる事前定義されたタイプのリストを含む。

50

【0071】

本開示の少なくとも1つの例によれば、ランタイムにプログラムのソースコードに対して行われた変更を実施するためのコンピューティングデバイスが提供される。コンピューティングデバイスは、プロセッサと、複数の命令が記憶されたメモリとを含み得、複数の命令は、プロセッサによって実行されるとき、コンピューティングデバイスに、ランタイム環境でソースコードを実行させ、ランタイム環境でランタイムにソースコードに対して行われた変更を検出させ、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定させ、リロード可能と識別されたソースコード内のタイプに対して変更が行われたという判定に応答して、タイプの新しいバージョンを作成させ、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行わせる。

10

【0072】

上記のコンピューティングデバイスの少なくとも1つの態様によれば、コンピューティングデバイスは、ランタイム環境でソースコードを実行することが、ソースコードを取得することと、統合開発環境（IDE）内にソースコードをロードすることと、IDEによって配置されたランタイム環境でソースコードを実行することとを含むことを含み得る。

【0073】

上記のコンピューティングデバイスの少なくとも1つの態様によれば、コンピューティングデバイスは、リロード可能と識別されたソースコード内のタイプに対して変更が行われなかったという判定に応答して、タイプをインプレースで更新するようにさらに構成され得る。

20

【0074】

上記のコンピューティングデバイスの少なくとも1つの態様によれば、コンピューティングデバイスは、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うことが、ソースコード内のタイプの前のバージョンのランタイム表現および1つまたは複数の既存のインスタンスを、タイプの新しいバージョンに置き換えることを含むことを含み得る。

【0075】

上記のコンピューティングデバイスの少なくとも1つの態様によれば、コンピューティングデバイスは、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うことが、ソースコードを構築するために使用されたフレームワークに、タイプの新しいバージョンが作成されたことを通知することと、ソースコード内のタイプの前のバージョンの1つまたは複数の既存のインスタンスから新しいバージョンに状態を移すことを含むことを含み得る。

30

【0076】

上記のコンピューティングデバイスの少なくとも1つの態様によれば、コンピューティングデバイスは、ランタイム環境内の変更を伴う更新後のソースコードの実行を続行するようにさらに構成され得る。

【0077】

上記のコンピューティングデバイスの少なくとも1つの態様によれば、コンピューティングデバイスは、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することが、ソースコードを構築するために使用されたフレームワークに関連付けられるフレームワークライブラリに基づいて、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定することを含み得、フレームワークライブラリは、リロード可能とマークされる事前定義されたタイプのリストを含むことを含み得る。

40

【0078】

本開示の少なくとも1つの例によれば、ランタイムにプログラムのソースコードに対して行われた変更を実施するための命令を記憶する非一時的コンピュータ可読媒体が提供される。命令は、コンピューティングデバイスの1つまたは複数のプロセッサによって実行

50

されるとき、コンピューティングデバイスに、ランタイム環境でソースコードを実行させ、ランタイム環境でランタイムにソースコードに対して行われた変更を検出させ、リロード可能と識別されたソースコード内のタイプに対して変更が行われたかどうかを判定させ、リロード可能と識別されたソースコード内のタイプに対して変更が行われたという判定に応答して、タイプの新しいバージョンを作成させ、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行わせる。

【0079】

上記の非一時的コンピュータ可読媒体の少なくとも1つの態様によれば、命令は、コンピューティングデバイスの1つまたは複数のプロセッサによって実行されるとき、ランタイム環境でソースコードを実行することが、ソースコードを取得することと、統合開発環境（IDE）内にソースコードをロードすることと、IDEによって配置されたランタイム環境でソースコードを実行することを含むことを含み得る。

10

【0080】

上記の非一時的コンピュータ可読媒体の少なくとも1つの態様によれば、命令は、1つまたは複数のプロセッサによって実行されるとき、コンピューティングデバイスに、リロード可能と識別されたソースコード内のタイプに対して変更が行われなかったという判定に応答して、タイプをインプレースで更新させ得る。

【0081】

上記の非一時的コンピュータ可読媒体の少なくとも1つの態様によれば、命令は、コンピューティングデバイスの1つまたは複数のプロセッサによって実行されるとき、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うことが、ソースコード内のタイプの前のバージョンのランタイム表現および1つまたは複数の既存のインスタンスを、タイプの新しいバージョンに置き換えることを含むことを含み得る。

20

【0082】

上記の非一時的コンピュータ可読媒体の少なくとも1つの態様によれば、命令は、コンピューティングデバイスの1つまたは複数のプロセッサによって実行されるとき、タイプの新しいバージョンを使用して、ランタイムにソースコードに対する変更の実施を行うことが、ソースコードを構築するために使用されたフレームワークに、タイプの新しいバージョンが作成されたことを通知し、ソースコード内のタイプの前のバージョンの1つまたは複数の既存のインスタンスから新しいバージョンに状態を移すことを含むことを含み得る。

30

【0083】

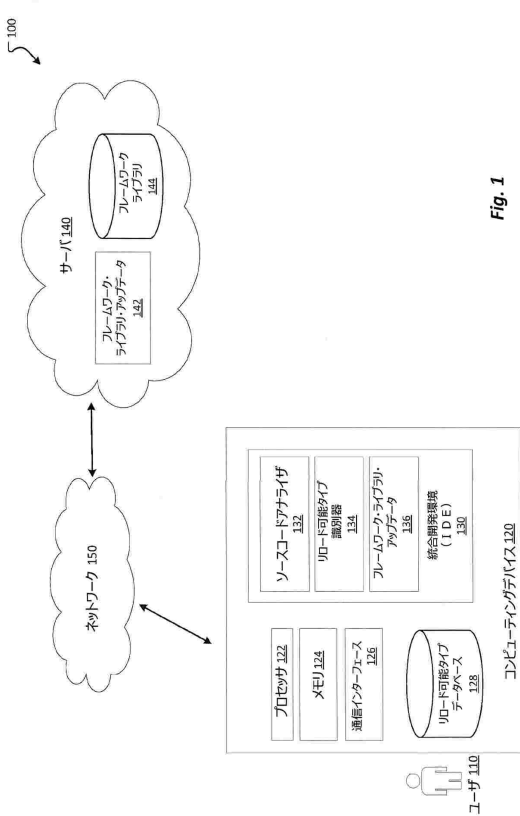
上記の非一時的コンピュータ可読媒体の少なくとも1つの態様によれば、命令はさらに、1つまたは複数のプロセッサによって実行されるとき、コンピューティングデバイスに、ランタイム環境内の変更を伴う更新後のソースコードの実行を続行させ得る。

【0084】

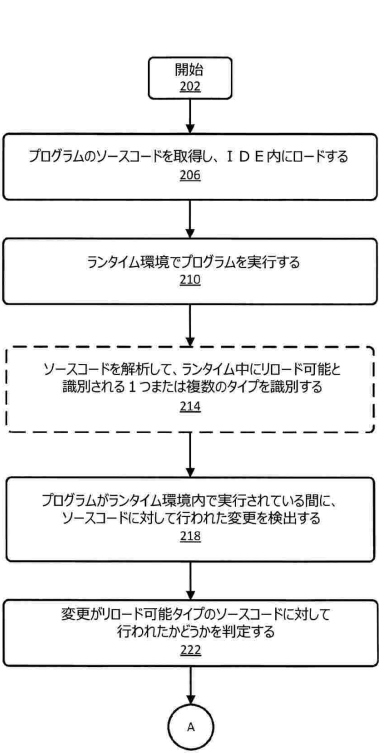
本開示は、様々な構成および態様において、実質的に本明細書で図示され、説明されたような構成要素、方法、プロセス、システム、および／または装置を含み、それらの様々な組合せ、サブコンビネーション、およびサブセットを含む。本開示を理解した後、本明細書で開示されるシステムおよび方法をどのように作成および使用するかを当業者は理解されよう。本開示は、様々な構成および態様において、本明細書で図示および／または説明されない項目を伴わずにデバイスおよびプロセスを提供することを含み、あるいは本明細書の様々な構成または態様において、たとえば実施の性能を改善し、容易さを達成し、および／またはコストを削減するために、前のデバイスまたはプロセスで使用された可能性のあるような項目を伴わないことを含む。

40

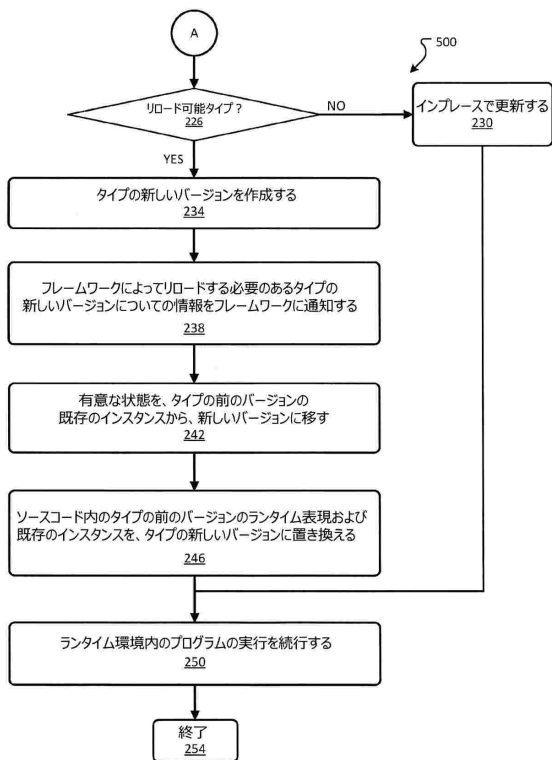
【図面】
【図 1】



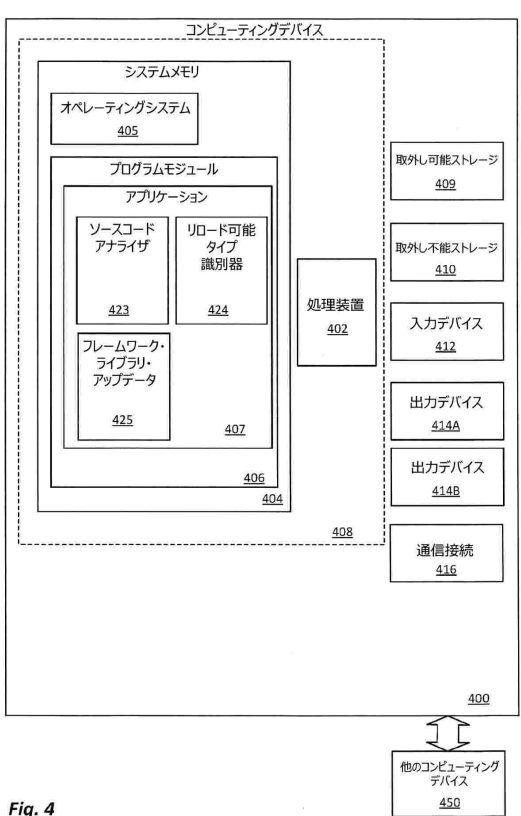
【図 2】



【図 3】



【図 4】



【図 5 A】

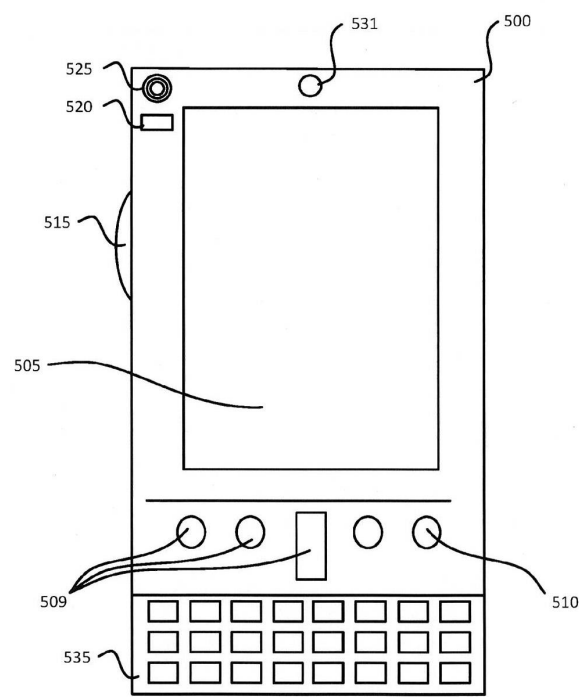


Fig. 5A

【図 5 B】

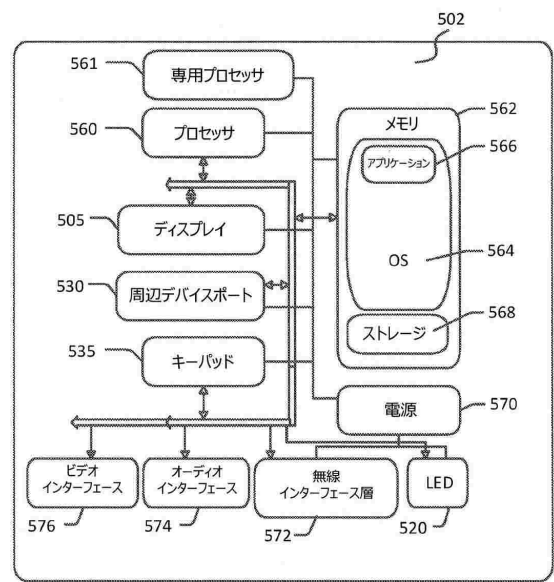


Fig. 5B

【図 6】

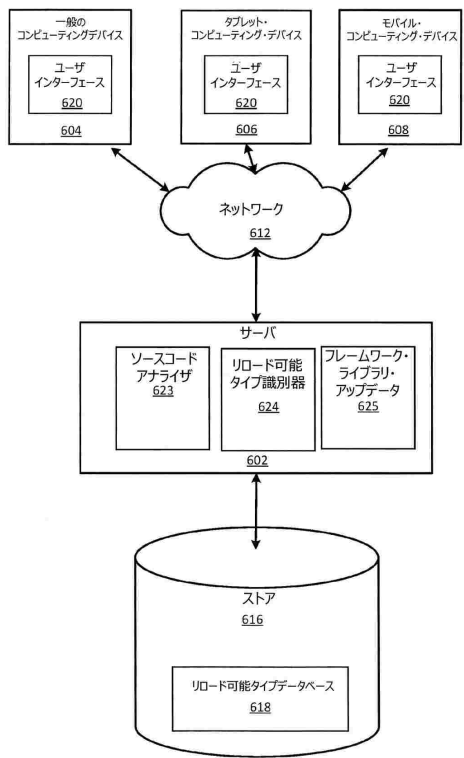


Fig. 6

10

20

30

40

50

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2022/035398

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F8/656 G06F8/71 G06F9/445
ADD.

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)
G06F

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 10 146 515 B1 (SUNDRESH SAMEER [US]) 4 December 2018 (2018-12-04) column 2, lines 60-62 column 4, lines 3-16 column 5, lines 30-39 column 6, lines 58-61 column 7, lines 32-39 column 8, lines 1-11 ----- -/--	1-15

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"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

"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

"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

"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

"&" document member of the same patent family

Date of the actual completion of the international search

12 October 2022

Date of mailing of the international search report

20/10/2022

Name and mailing address of the ISA/
European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Roux-Decker, D

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2022/035398

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	RAUDJARV REIN ET AL: "JRebel.Android: Runtime Class- and Resource Reloading for Android", 2015 IEEE/ACM 37TH IEEE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, IEEE, vol. 2, 16 May 2015 (2015-05-16), pages 741-744, XP033195150, DOI: 10.1109/ICSE.2015.337 [retrieved on 2015-08-14] page 741, column 2, lines 1-5 II. Approach; page 741, column 2, lines 3-12 page 742, column 1, lines 35-44 page 742, column 1, lines 48-52 -----	1-15

10

20

30

40

50

INTERNATIONAL SEARCH REPORT				International application No PCT/US2022/035398	
Information on patent family members					
Patent document cited in search report		Publication date	Patent family member(s)		Publication date
US 10146515	B1	04-12-2018	US	10146515 B1	04-12-2018
			US	10146522 B1	04-12-2018
			US	10795660 B1	06-10-2020

フロントページの続き

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,N
E,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,
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,J
O,JP,KE,KG,KH,KN,KP,KR,KW,KZ,LA,LC,LK,LR,LS,LU,LY,MA,MD,ME,MG,MK,MN,MW,MX,MY,M
Z,NA,NG,NI,NO,NZ,OM,PA,PE,PG,PH,PL,PT,QA,RO,RS,RU,RW,SA,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. VISUAL BASIC
- 2. PYTHON
- 3. HDMI
- 4. JAVASCRIPT

クロソフト ウェイ, マイクロソフト テクノロジー ライセンシング, エルエルシー

Fターム (参考) 5B042 HH39 HH48
5B376 DA05