

(19) 日本国特許庁(JP)

(12) 公表特許公報(A)

(11) 特許出願公表番号

特表2019-523918

(P2019-523918A)

(43) 公表日 令和1年8月29日(2019.8.29)

(51) Int.Cl.	F I	テーマコード (参考)
<b>G06F 9/451 (2018.01)</b>	G06F 9/451	5B376
<b>G10L 15/28 (2013.01)</b>	G10L 15/28 200	5L049
<b>G10L 15/22 (2006.01)</b>	G10L 15/22 300U	
<b>G10L 15/30 (2013.01)</b>	G10L 15/30	
<b>G06F 3/16 (2006.01)</b>	G06F 3/16 650	

審査請求 未請求 予備審査請求 未請求 (全 39 頁) 最終頁に続く

(21) 出願番号 特願2018-559707 (P2018-559707)  
 (86) (22) 出願日 平成29年5月10日 (2017.5.10)  
 (85) 翻訳文提出日 平成31年1月10日 (2019.1.10)  
 (86) 国際出願番号 PCT/US2017/032002  
 (87) 国際公開番号 W02017/197010  
 (87) 国際公開日 平成29年11月16日 (2017.11.16)  
 (31) 優先権主張番号 62/334, 434  
 (32) 優先日 平成28年5月10日 (2016.5.10)  
 (33) 優先権主張国・地域又は機関  
 米国 (US)  
 (31) 優先権主張番号 62/336, 566  
 (32) 優先日 平成28年5月13日 (2016.5.13)  
 (33) 優先権主張国・地域又は機関  
 米国 (US)

(71) 出願人 502208397  
 グーグル エルエルシー  
 アメリカ合衆国 カリフォルニア州 94  
 043 マウンテン ビュー アンフィシ  
 アター パークウェイ 1600  
 (74) 代理人 110001195  
 特許業務法人深見特許事務所  
 (72) 発明者 ミクスター, ケネス  
 アメリカ合衆国、94043 カリフォル  
 ニア州、マウンテン・ビュー、アンフィシ  
 アター・パークウェイ、1600  
 Fターム(参考) 5B376 FA25 GA03 GA11  
 5L049 CC12

最終頁に続く

(54) 【発明の名称】 デバイス上の音声アシスタントの実装

## (57) 【要約】

オーディオ入力システムを有する電子デバイスにおける方法は、デバイスにおいて口頭入力を受信するステップと、口頭入力を処理するステップと、口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信するステップと、口頭入力に基づいた情報に応じて遠隔システムが生成した、要求に対する応答を受信するステップと、応答に応じた動作を実行するステップとを含み、前記した、受信するステップ、処理するステップ、送信するステップ、受信するステップ、および実行するステップのうちの1つ以上は、電子デバイス上で実行中の音声アシスタントライブラリの1つ以上の音声処理モジュールによって実行され、音声処理モジュールは、電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する。

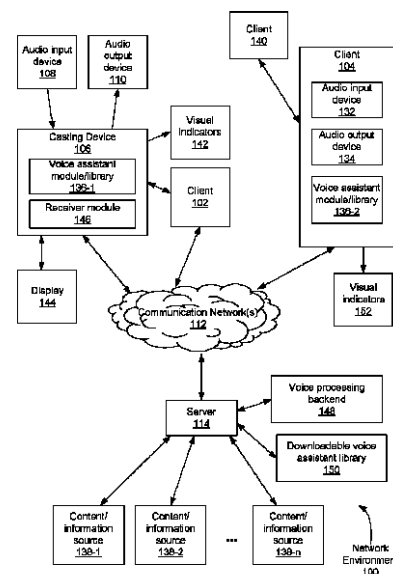


FIGURE 1

**【特許請求の範囲】****【請求項 1】**

オーディオ入力システムと、1つ以上のプロセッサと、前記1つ以上のプロセッサによって実行される1つ以上のプログラムを格納したメモリとを備える電子デバイスにおいて、

前記デバイスにおいて口頭入力を受信するステップと、

前記口頭入力を処理するステップと、

前記口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信するステップと、

前記口頭入力に基づいた情報に応じて前記遠隔システムが生成した、前記要求に対する応答を受信するステップと、

前記応答に応じた動作を実行するステップとを含み、

前記受信するステップ、前記処理するステップ、前記送信するステップ、前記受信するステップ、および前記実行するステップのうちの1つ以上は、前記電子デバイス上で実行中の音声アシスタントライブラリの1つ以上の音声処理モジュールによって実行され、前記音声処理モジュールは、前記電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する、方法。

**【請求項 2】**

前記音声処理モジュールに関連付けられた少なくともいくつかの音声処理動作は、ワイドエリアネットワークを介して前記電子デバイスと互いに接続される前記遠隔システム上で実行される、請求項1に記載の方法。

**【請求項 3】**

前記音声アシスタントライブラリは、複数の異なるデバイスタイプ上で動作可能な共通のオペレーティングシステム上で実行可能であり、それによって、前記音声処理動作のうちの1つ以上と対話するように構成された音声対応アプリケーションのポータビリティを可能にする、先行する請求項のいずれか1項に記載の方法。

**【請求項 4】**

オーディオ入力システムを備える電子デバイス用の、デバイスを問わない音声アシスタントライブラリであって、

複数の異なる電子デバイスタイプ上に実装される共通のオペレーションシステム上で動作するように構成された1つ以上の音声処理モジュールを含み、前記音声処理モジュールは、前記電子デバイス上で実行中のアプリケーションプログラムおよびオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供し、それによって、前記音声処理動作のうちの1つ以上と対話するように構成された音声対応アプリケーションのポータビリティが可能になる、音声アシスタントライブラリ。

**【請求項 5】**

前記音声処理モジュールに関連付けられた少なくともいくつかの音声処理動作は、ワイドエリアネットワークを介して前記電子デバイスと互いに接続されるバックエンドサーバ上で実行される、先行する請求項のいずれか1項に記載の音声アシスタントライブラリ。

**【請求項 6】**

前記音声処理動作は、前記電子デバイスに接続されたデバイスを制御するように構成されたデバイス固有の動作を含む、先行する請求項のいずれか1項に記載の音声アシスタントライブラリ。

**【請求項 7】**

前記音声処理動作は、要求された情報および/またはメディアコンテンツを前記電子デバイスのユーザに提供するまたは前記電子デバイスに接続されたデバイス上で提供するように構成された情報/メディア要求動作を含む、先行する請求項のいずれか1項に記載の音声アシスタントライブラリ。

**【請求項 8】**

10

20

30

40

50

オーディオ入力システムと、  
1つ以上のプロセッサと、

前記1つ以上のプロセッサによって実行される1つ以上のプログラムを格納したメモリとを備え、前記1つ以上のプログラムは、

前記デバイスにおいて口頭入力を受信するための命令と、

前記口頭入力を処理するための命令と、

前記口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信するための命令と、

前記口頭入力に基づいた情報に応じて前記遠隔システムが生成した、前記要求に対する応答を受信するための命令と、

前記応答に応じた動作を実行するための命令を含み、

前記受信すること、前記処理すること、前記送信すること、前記受信すること、および前記実行することのうちの1つ以上は、前記電子デバイス上で実行中の前記音声アシスタントライブラリの前記1つ以上の音声処理モジュールによって実行され、前記音声処理モジュールは、前記電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する、電子デバイス。

【請求項9】

前記音声処理モジュールに関連付けられた少なくともいくつかの音声処理動作は、ワイドエリアネットワークを介して前記電子デバイスと互いに接続される前記遠隔システム上で実行される、請求項8に記載のデバイス。

【請求項10】

前記音声アシスタントライブラリは、複数の異なるデバイスタイプ上で動作可能な共通のオペレーティングシステム上で実行可能であり、それによって、前記音声処理動作のうちの1つ以上と対話するように構成された音声対応アプリケーションのポータビリティを可能にする、先行する請求項のいずれか1項に記載のデバイス。

【請求項11】

1つ以上のプログラムを格納した非一時的なコンピュータ読み取り可能な記憶媒体であって、前記1つ以上のプログラムは、命令を含み、前記命令は、オーディオ入力システムと1つ以上のプロセッサとを有する電子デバイスによって実行されると、前記電子デバイスに、

前記デバイスにおいて口頭入力を受信させ、

前記口頭入力を処理させ、

前記口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信させ、

前記口頭入力に基づいた情報に応じて前記遠隔システムが生成した、前記要求に対する応答を受信させ、

前記応答に応じた動作を実行させ、

前記受信すること、前記処理すること、前記送信すること、前記受信すること、および前記実行することのうちの1つ以上は、前記電子デバイス上で実行中の前記音声アシスタントライブラリの前記1つ以上の音声処理モジュールによって実行され、前記音声処理モジュールは、前記電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する、非一時的なコンピュータ読み取り可能な記憶媒体。

【請求項12】

前記音声処理モジュールに関連付けられた少なくともいくつかの音声処理動作は、ワイドエリアネットワークを介して前記電子デバイスと互いに接続される前記遠隔システム上で実行される、請求項11に記載のコンピュータ読み取り可能な記憶媒体。

【請求項13】

前記音声アシスタントライブラリは、複数の異なるデバイスタイプ上で動作可能な共通のオペレーティングシステム上で実行可能であり、それによって、前記音声処理動作のう

10

20

30

40

50

ちの１つ以上と対話するように構成された音声対応アプリケーションのポータビリティを可能にする、先行する請求項のいずれか１項に記載のコンピュータ読み取り可能な記憶媒体。

【請求項１４】

オーディオ入力システムと、  
１つ以上のプロセッサと、

前記１つ以上のプロセッサによって実行される１つ以上のプログラムを格納したメモリとを備え、前記１つ以上のプログラムは、請求項１～３のいずれか１項に記載の方法を実行するための命令を含む、電子デバイス。

【請求項１５】

１つ以上のプログラムを格納した非一時的なコンピュータ読み取り可能な記憶媒体であって、前記１つ以上のプログラムは、命令を含み、前記命令は、オーディオ入力システムと１つ以上のプロセッサとを有する電子デバイスによって実行されると、前記電子デバイスに、請求項１～３のいずれか１項に記載の方法を実行させる、非一時的なコンピュータ読み取り可能な記憶媒体。

【発明の詳細な説明】

【技術分野】

【０００１】

技術分野

本願は、一般に、これに限定されないが、デバイスおよび関連ライブラリ用の音声アシスタントを含むコンピュータ技術に関する。

【背景技術】

【０００２】

背景

インターネットおよびクラウドコンピューティングの発展と並んで、オーディオ／音声入出力を通してユーザと対話する音声ベースのアシスタントの人気の高まっている。これらのアシスタントは、デジタルメディアを消費するためのインタフェースを提供し、一例を挙げると、ニュース、スポーツの得点、天気、および株といった様々な種類の情報を提供できる。

【０００３】

ユーザは、音声ベースのアシスタント機能があれば望ましいデバイスを複数有しているだろう。いろいろなデバイス間で実装および使用でき、これらいろいろなデバイス間で一貫性のあるエクスペリエンスを提供でき、かつ、特定のデバイスに特化した機能をサポートできる、音声ベースのアシスタントを有することが望ましい。

【発明の概要】

【課題を解決するための手段】

【０００４】

概要

本明細書に記載の実装形態は、多種多様なオペレーティングシステムプラットフォームのためにローカルデバイスを制御することを可能にする方法で音声アシスタントを組み込みシステムおよび／またはデバイスに組込むまたは含むことを対象とする。

【０００５】

いくつかの実装形態によれば、薄くてリソースの使用量が少ないデバイス側ライブラリは、オーディオデータのローカル処理、ウェイクワードまたはホットワードのリスニング、およびユーザ要求の送信を含む特徴がある。さらなる特徴として、クラウド脳への接続性、拡張可能な音声操作制御システム、多くの多様な動作環境への統合を可能にするポータビリティレイヤ、および残りのクライアントソフトウェアと非同期で更新できることなどがある。

【０００６】

記載の実装形態には、多くの異なるデバイス間で音声アシスタントと対話するための同

10

20

30

40

50

じようなユーザエクスペリエンスを提供するという利点がある。

【0007】

記載の実装形態には、音声アシスタント機能におけるイノベーションを、デバイス自体から使用可能なイノベーションと切り離すことが可能になるという別の利点がある。たとえば、改良された認識パイプラインが作られた場合、認識結果がデバイスに出力される一方で、デバイスのメーカーは、認識結果を受信するために何かをする必要なしに、引き続き、前の音声コマンドから利益を得ることができる。

【0008】

いくつかの実装形態によれば、オーディオ入力システムと、1つ以上のプロセッサと、1つ以上のプロセッサによって実行される1つ以上のプログラムを格納したメモリとを有する電子デバイスにおける方法は、デバイスにおいて口頭入力を受信するステップと、口頭入力を処理するステップと、口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信するステップと、口頭入力に基づいた情報に応じて遠隔システムが生成した、要求に対する応答を受信するステップと、応答に応じた動作を実行するステップとを含み、受信するステップ、処理するステップ、送信するステップ、受信するステップ、および実行するステップのうちの1つ以上は、電子デバイス上で実行中の音声アシスタントライブラリの1つ以上の音声処理モジュールによって実行され、音声処理モジュールは、電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する。

【0009】

いくつかの実装形態では、オーディオ入力システムを備える電子デバイス用の、デバイスを問わない音声アシスタントライブラリは、複数の異なる電子デバイスタイプ上に実装される共通のオペレーションシステム上で動作するように構成された1つ以上の音声処理モジュールを含み、当該音声処理モジュールは、電子デバイス上で実行中のアプリケーションプログラムおよびオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供しそれによって、音声処理動作のうちの1つ以上と対話するように構成された音声対応アプリケーションのポータビリティが可能なる。

【0010】

いくつかの実装形態では、電子デバイスは、オーディオ入力システムと、1つ以上のプロセッサと、1つ以上のプロセッサによって実行される1つ以上のプログラムを格納したメモリとを備える。1つ以上のプログラムは、デバイスにおいて口頭入力を受信するための命令と、口頭入力を処理するための命令と、口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信するための命令と、口頭入力に基づいた情報に応じて遠隔システムが生成した、要求に対する応答を受信するための命令と、応答に応じた動作を実行するための命令を含み、受信すること、処理すること、送信すること、受信すること、および実行することのうちの1つ以上は、電子デバイス上で実行中の音声アシスタントライブラリの1つ以上の音声処理モジュールによって実行され、音声処理モジュールは、電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する。

【0011】

いくつかの実装形態では、非一時的なコンピュータ読み取り可能な記憶媒体は、1つ以上のプログラムを格納する。1つ以上のプログラムは、命令を含み、命令は、オーディオ入力システムと1つ以上のプロセッサとを有する電子デバイスによって実行されると、電子デバイスに、デバイスにおいて口頭入力を受信させ、口頭入力を処理させ、口頭入力に基づいて決定された情報を含む要求を遠隔システムに送信させ、口頭入力に基づいた情報に応じて遠隔システムが生成した、要求に対する応答を受信させ、応答に応じた動作を実行させ、受信すること、処理すること、送信すること、受信すること、および実行することのうちの1つ以上は、電子デバイス上で実行中の音声アシスタントライブラリの1つ以上の音声処理モジュールによって実行され、音声処理モジュールは、電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーテ

10

20

30

40

50

イングソフトウェアにアクセス可能な複数の音声処理動作を提供する。

【図面の簡単な説明】

【0012】

【図1】いくつかの実装形態に係る、ネットワーク環境の例を示すブロック図である。

【図2】いくつかの実装形態に係る、音声アシスタントクライアントデバイスの例を示す図である。

【図3】いくつかの実装形態に係る、サーバシステムの例を示す図である。

【図4】いくつかの実装形態に係る、音声アシスタントライブラリの機能ビューを示すブロック図である。

【図5】いくつかの実装形態に係る、デバイス上での口頭入力进行处理するための方法のフロー図である。

10

【発明を実施するための形態】

【0013】

図面の全体にわたって、同じ参照番号は、対応する部分を指す。

実装形態の説明

ここで、様々な実装形態を詳細に説明する。これらの実装形態の例は、添付の図面に示される。以下の詳細な説明において、本発明および記載の実装形態の十分な理解を与えるために、たくさんの具体的な詳細を説明する。しかしながら、これらの具体的な詳細がなくても、本発明を実施することはできる。その他の場合、周知の方法、プロシージャ、構成要素、および回路については、実装形態の態様を不必要にあいまいにしないよう、詳細に説明しない。

20

【0014】

いくつかの実装形態では、音声アシスタントの目的は、いろいろなデバイス間で利用可能であり、かつ、多種多様なユースケースを可能する、個人向けにカスタマイズされた音声インタフェースをユーザに提供することであり、ユーザの一日にわたって一貫したエクスペリエンスを提供する。音声アシスタントおよび/または関連機能は、ファーストパーティ製品およびデバイス、ならびにサードパーティ製品およびデバイスに統合されてもよい。

【0015】

ユースケース例は、メディアを含む。音声コマンドを用いて、音楽、ラジオ、ポッドキャスト、ニュース、およびその他のオーディオメディアの再生および制御を音声によって開始してもよい。たとえば、ユーザは、（たとえば、「ジャズ音楽を再生して」、「FM 107.5を再生して」、「次の曲にスキップして」、「“連続”再生して」）という音声コマンドを発して、様々な種類のオーディオメディアを再生または制御できる。さらには、このようなコマンドを用いて、地上波のラジオ放送局のオンラインストリーミング、音楽サブスクリプションサービス、ローカルストレージ、リモートストレージなど、いろいろなソースからのオーディオメディアを再生してもよい。さらには、音声アシスタントは、キャストデバイスで利用できる統合を利用して、さらなるコンテンツをサポートしてもよい。

30

【0016】

別のユースケース例は、リモート再生を含む。ユーザは、音声アシスタント機能を含むキャストデバイスに対して音声コマンドを発行してもよく、音声コマンドに応じて、コマンドにおいて指定されたデバイス上で、指定された1つ以上のデバイスからなる群に含まれるデバイス上で、または、コマンドにおいて指定されたエリアにある1つ以上のデバイス上で、メディアが再生される（たとえば、キャストされる）。また、ユーザは、一般的なカテゴリまたは特定のコンテンツをコマンドにおいて指定でき、コマンドにおいて指定されたカテゴリまたはコンテンツに応じて、適切なメディアが再生される。

40

【0017】

さらに別のユースケース例は、生産性を向上させる機能（たとえば、タイマ、目覚まし時計、カレンダー）、ホームオートメーション、検索エンジン（たとえば、検索クエリ）

50

の技術が活かされた質問と回答、楽しみ（たとえば、アシスタントの個性、ジョーク、ゲーム、イースターエッグ）、および日々のタスク（たとえば、交通手段、移動、食べ物、ファイナンス、贈り物など）などの非メディアである。

【0018】

いくつかの実装形態では、音声アシスタントは、キャストリングデバイスのオプション機能として提供され、音声アシスタント機能は、キャストリングデバイスの一部として更新されてもよい。

【0019】

いくつかの実装形態では、音声コマンドおよびユーザからの口頭入力に含まれるホットワードまたはキーワードの検出は、アプリケーションプロセッサによって行われる（たとえば、ユーザが音声コマンドまたは口頭入力を話しかける先のクライアントデバイスまたはキャストリングデバイスにおいて行われる）。いくつかの実装形態では、ホットワードの検出は、外部のデジタル信号プロセッサによって行われる（ユーザが音声コマンドまたは口頭入力を話しかける先のクライアントデバイスまたはキャストリングデバイスとは対照的に、たとえば、サーバシステムが音声コマンドを処理することによって行われる）。

【0020】

いくつかの実装形態では、音声アシスタント機能を有するデバイスは、遠方界サポート、「プッシュ・トゥ・アシスト」または「プッシュ・トゥ・トーク」（たとえば、音声アシスタント機能を開始するためのボタン）、およびAC電源のうちの1つ以上を含む。

【0021】

いくつかの実装形態では、音声アシスタントは、オーディオ入力装置（たとえば、マイクロホン、進行中の再生のメディアループバック）、マイクロホンの状態（たとえば、オン/オフ）、ダッキング（たとえば、ホットワードまたはプッシュ・トゥ・トークによってアシスタントが起動（トリガー：trigger）されたときにすべての出力の音量を下げる）、ならびに新しいアシスタントイベントおよびステータスメッセージ（たとえば、アシスタントが起動された（たとえば、ホットワードを聞いた、アシスタントボタンが押された）、音声をリスニングする、サーバ上で待機する、応答する、応答が終了する、アラーム/タイマが鳴っている）のうちの1つ以上のためのアプリケーションプログラミングインタフェース（API）を含む。

【0022】

いくつかの実装形態では、音声アシスタント機能を有するデバイスは、デバイス上の音声アシスタントの機能を可能にまたは容易にする（たとえば、デバイス上の音声アシスタント機能をセットアップする、ユーザにチュートリアルを提供する）ために、設定上の目的のために別のデバイスと（たとえば、スマートフォン上の設定アプリケーションと）通信してもよい。設定またはセットアップには、デバイスの位置の指定、ユーザアカウントとの関連付け、ユーザの音声制御へのオプトイン、メディアサービス（たとえば、映像ストリーミングサービス、音楽ストリーミングサービス）へリンクすることおよびメディアサービスの優先順位をつけること、ホームオートメーション設定などが含まれてもよい。

【0023】

いくつかの実装形態では、音声アシスタントを有するデバイスは、1つ以上のユーザインタフェース要素またはユーザに対する表示を含んでもよい。ユーザインタフェース要素のうちの1つ以上は、物理的要素であり（たとえば、1つ以上のLEDを用いて表示される光のパターン、スピーカが出力するサウンドパターン）、ホットワードに左右されない「プッシュ・トゥ・アシスト」または「プッシュ・トゥ・トーク」トリガー、「ミュートマイクロホン」トリガーおよび視覚的なステータス表示、「ホットワード待ちステータス」の視覚的な表示、「ホットワードを検出」の視覚的な表示、少し離れた位置（たとえば、15フィート）から視認できる「アシスタントは積極的にリスニング中です」の視覚的な表示、「アシスタントが作業中/考え中」の視覚的な表示、「音声メッセージ/通知があります」の視覚的な表示、「音量レベル」の制御方法およびステータスインジケータ、ならびに「一旦停止/再開」制御方法のうちの1つ以上を含んでもよい。いくつかの実装

10

20

30

40

50

形態では、これらの物理的なユーザインタフェース要素は、クライアントデバイスまたはキャストデバイスによって提供される。いくつかの実装形態では、音声アシスタントは、エクスペリエンスが異なるデバイス間で一貫するように、異なるデバイス間で共通のユーザインタフェース要素または表示のセットをサポートする。

【0024】

いくつかの実装形態では、音声アシスタントは、デバイス固有のコマンドおよび/またはホットワード、ならびに、コマンドおよび/またはホットワードの定義済みの標準セットをサポートする。

【0025】

図1は、いくつかの実装形態に係る、ネットワーク環境100を示す図である。ネットワーク環境100は、キャストデバイス106および/または音声アシスタントクライアントデバイス104を含む。キャストデバイス106（たとえば、GOOGLE INC.によるCHROMECAST）は、オーディオ入力装置108（たとえば、マイクロホン）およびオーディオ出力装置110（たとえば、1つ以上のスピーカ）に直接または通信可能に接続される。いくつかの実装形態では、オーディオ入力装置108およびオーディオ出力装置110は、キャストデバイス106に通信可能に接続されたデバイス（たとえば、スピーカシステム、テレビ、サウンドバー）のコンポーネントである。いくつかの実装形態では、オーディオ入力装置108は、キャストデバイス106のコンポーネントであり、オーディオ出力装置110は、キャストデバイス106が通信可能に接続されたデバイスのコンポーネントであり、または、オーディオ出力装置110は、キャストデバイス106のコンポーネントであり、オーディオ入力装置108は、キャストデバイス106が通信可能に接続されたデバイスのコンポーネントである。いくつかの実装形態では、オーディオ入力装置108およびオーディオ出力装置110は、キャストデバイス106のコンポーネントである。

【0026】

いくつかの実装形態では、キャストデバイス106は、クライアント102に通信可能に接続される。クライアント102は、音声アシスタント機能を含む、キャストデバイス106の構成を容易にするアプリケーションまたはモジュール（たとえば、キャストデバイス設定アプリ）を含んでもよい。

【0027】

いくつかの実装形態では、キャストデバイス106は、ディスプレイ144に接続される。

【0028】

いくつかの実装形態では、キャストデバイス106は、1つ以上の視覚的インジケータ142（たとえば、LEDライト）を含む。

【0029】

いくつかの実装形態では、キャストデバイス106は、受信モジュール146を含む。いくつかの実装形態では、受信モジュール146は、キャストデバイス106を操作し、操作対象として、たとえば、ハードウェア関数およびコンテンツソースとの通信を含む。いくつかの実装形態では、キャストデバイス106において、異なるコンテンツソースに対して異なる受信モジュール146がある。いくつかの実装形態では、受信モジュール146は、異なるコンテンツソースのためのサブモジュールをそれぞれ含む。

【0030】

音声アシスタントクライアントデバイス104（たとえば、GOOGLE INC.によるGOOGLEアシスタント、GOOGLE INC.によるGOOGLE HOMEを有するスマートフォン、ラップトップもしくはデスクトップコンピュータ、タブレットコンピュータ、音声コマンドデバイス、モバイル機器、または車載システム）は、オーディオ入力装置132（たとえば、マイクロホン）と、オーディオ出力装置134（たとえば、1つ以上のスピーカ、ヘッドフォン）とを備える。いくつかの実装形態では、音声ア

10

20

30

40

50



シスタントクライアントデバイス 104 (たとえば、GOOGLE INC. によるGOOGLE アシスタント、GOOGLE INC. によるGOOGLE HOME を有する音声コマンドデバイス、モバイル機器、または車載システム) は、クライアント 140 (たとえば、スマートフォン、タブレットデバイス) に通信可能に接続される。クライアント 140 は、音声アシスタント機能を含む、音声アシスタントクライアントデバイス 104 の設定を容易にするアプリケーションまたはモジュール (たとえば、音声コマンドデバイス設定アプリ) を含んでもよい。

#### 【0031】

いくつかの実装形態では、音声アシスタントクライアントデバイス 104 は、1 つ以上の視覚的インジケータ 152 (たとえば、LED ライト) を含む。視覚的なインジケータ (たとえば、LED ライト) を有する音声アシスタントクライアントデバイスの例を、2016 年 5 月 13 日に出版され、「LED Design Language for Visual Affordance of Voice User Interfaces (音声ユーザインタフェースの視覚的アフォーダンスのための LED デザイン言語)」と題された米国仮出願第 62/336,566 号 (引用により本明細書に援用する) を示した図 4A に示す。

#### 【0032】

キャストिंगデバイス 106 および音声アシスタントクライアントデバイス 104 は、音声アシスタントモジュールまたはライブラリ 136 のそれぞれのインスタンスを含む。音声アシスタントモジュール/ライブラリ 136 は、いろいろなデバイス (たとえば、キャストिंगデバイス 106、音声アシスタントクライアントデバイス 104) 間で音声アシスタント機能を実装するモジュール/ライブラリである。音声アシスタント機能は、デバイス固有の特徴 (たとえば、デバイス固有の特徴を音声アシスタントによって制御するためのサポート) を引き続き許可しつつ、いろいろなデバイス間で一貫性がある。いくつかの実装形態では、音声アシスタントモジュール/ライブラリ 136 は、デバイス間で同じまたは同様であり、同じライブラリのインスタンスがいろいろなデバイスに含まれる。

#### 【0033】

いくつかの実装形態では、デバイスのタイプによっては、音声アシスタントモジュール/ライブラリ 136 は、デバイスにインストールされたアプリケーションにもしくはデバイスのオペレーティングシステムに含まれる、またはデバイスに埋め込まれる (たとえば、ファームウェアに埋め込まれる)。

#### 【0034】

いくつかの実装形態では、キャストिंगデバイス 106 における音声アシスタントモジュール/ライブラリ 136 - 1 は、受信モジュール 146 と通信して、音声アシスタント動作を行う。

#### 【0035】

いくつかの実装形態では、キャストिंगデバイス 106 における音声アシスタントモジュール/ライブラリ 136 - 1 は、視覚的インジケータ 142 を制御できる、または視覚的インジケータ 142 に影響を与えることができる。

#### 【0036】

いくつかの実装形態では、音声アシスタントクライアントデバイス 104 における音声アシスタントモジュール/ライブラリ 136 - 2 は、視覚的インジケータ 152 を制御できる、または視覚的インジケータ 152 に影響を与えることができる。

#### 【0037】

キャストिंगデバイス 106 および音声アシスタントクライアントデバイス 104 は、1 つ以上の通信ネットワーク 112 (たとえば、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、インターネット) を通じてサーバシステム 114 に通信可能に接続される。音声アシスタントモジュール/ライブラリ 136 は、オーディオ入力装置 108 / 132 によって拾われた (たとえば、キャプチャされた) 口頭入力を検出 (たとえば、受信) し、当該口頭入力を (たとえば、ホットワードを検出するために) 処理し、処理済

10

20

30

40

50

みの口頭入力または処理済みの口頭入力をエンコーディングしたものをサーバ 114 に送信する。サーバ 114 は、処理済みの口頭入力またはそれをエンコーディングしたものを受信し、受信した口頭入力を処理して、当該口頭入力に対する適切な応答を決定する。適切な応答は、コンテンツ、情報、または、キャストデバイス 106 または音声アシスタントクライアントデバイス 104 に関数または動作を実行させるためのキャストデバイス 106 または音声アシスタントクライアントデバイス 104 に対する命令、コマンド、もしくはメタデータであってもよい。サーバ 114 は、コンテンツもしくは情報が出力される（たとえば、オーディオ出力装置 110 / 134 から出力される）および / または関数を実行されるキャストデバイス 106 または音声アシスタントクライアントデバイス 104 に、応答を送る。処理の一部として、サーバ 114 は、1 つ以上のコンテンツ / 情報ソース 138 と通信し、応答用に、コンテンツまたは情報を取得するまたはそのようなものを参照してもよい。いくつかの実装形態では、コンテンツ / 情報ソース 138 として、検索エンジン、データベース、ユーザのアカウントに関連付けられた情報（たとえば、カレンダー、タスク一覧、電子メール）、ウェブサイト、およびメディアストリーミングサービスなどが挙げられる。いくつかの実装形態では、音声アシスタントクライアントデバイス 104 およびキャストデバイス 106 は、互いに通信または対話を行ってもよい。このような通信または対話の例、および音声アシスタントクライアントデバイス 104（たとえば、GOOGLE INC. による GOOGLE HOME）の動作の例が、2016 年 5 月 13 日に出願され、「LED Design Language for Visual Affordance of Voice User Interfaces（音声ユーザインタフェースの視覚的アフォーダンスについてのデザイン言語）」と題された米国仮出願第 62 / 336, 566 号、2016 年 5 月 13 日に出願され、「Voice-Controlled Closed Caption Display（音声制御されたクロズドキャプションの表示）」と題された米国仮出願第 62 / 336, 569 号、および 2016 年 5 月 13 日に出願され、「Media Transfer among Media Output Devices（メディア出力デバイス間のメディア転送）」と題された米国仮出願第 62 / 336, 565 号に開示されている。これらの出願のすべては、引用により本明細書に援用する。

10

20

30

40

50

#### 【0038】

いくつかの実装形態では、音声アシスタントモジュール / ライブラリ 136 は、オーディオ入力装置 108 / 132 によってキャプチャされた口頭入力を受信し、当該口頭入力（処理をせずまたはほとんどせずに）またはそれをエンコーディングしたものをサーバ 114 に送信する。サーバ 114 は、口頭入力を処理して、ホットワードを検出し、適切な応答を決定し、この応答をキャストデバイス 106 または音声アシスタントクライアントデバイス 104 に送る。

#### 【0039】

キャストデバイス 106 または音声アシスタントクライアントデバイス 104 が関数を実行するためのコマンドを当該口頭入力を含むとサーバ 114 が判断した場合、サーバ 114 は、キャストデバイス 106 または音声アシスタントクライアントデバイス 104 に当該関数を実行するように指示する命令またはメタデータを応答に含めて送信する。当該関数は、デバイスに固有であってもよく、音声アシスタントにおけるこのような関数をサポートするための機能が、音声アシスタントモジュール / ライブラリ 136 に追加またはリンクされるカスタムモジュールまたは関数として、キャストデバイス 106 またはクライアント 104 に含まれてもよい。

#### 【0040】

いくつかの実装形態では、サーバ 114 は、口頭入力の処理動作を行い、かつ、当該口頭入力に対する応答を決定する音声処理バックエンド 148 を含む、またはこれに接続される。

#### 【0041】

いくつかの実装形態では、サーバ 114 は、ダウンロード可能な音声アシスタントライブラリ 150 を含む。ダウンロード可能な音声アシスタントライブラリ 150（たとえば

、音声アシスタントライブラリ 136 と同じまたはそれを更新したもの）は、新しい特徴もしくは機能、または更新を含んでもよく、デバイスに音声アシスタントライブラリを追加するまたは音声アシスタントライブラリ 136 を更新するためにダウンロードすることができる。

#### 【0042】

図 2 は、いくつかの実装形態に係る、ネットワーク環境 100 の音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 の例を示すブロック図である。音声アシスタントクライアントデバイス 104 の例として、携帯電話、タブレットコンピュータ、ラップトップコンピュータ、デスクトップコンピュータ、ワイヤレススピーカ（たとえば、GOOGLE INC. による GOOGLE HOME）、音声コマンドデバイス（たとえば、GOOGLE INC. による GOOGLE HOME）、テレビ、サウンドバー、キャストデバイス（たとえば、GOOGLE INC. による CHROMECAST）、メディアストリーミングデバイス、家電、家庭用電子機器、車載システム、およびウェアラブルパーソナルデバイスなどが挙げられるが、これらに限定されない。音声アシスタントクライアントデバイス 104（たとえば、GOOGLE INC. による GOOGLE HOME、GOOGLE アシスタント機能を有するモバイル機器）またはキャストデバイス 106（たとえば、GOOGLE INC. による CHROMECAST）は、通常、1 つ以上の処理装置（CPU）202 と、1 つ以上のネットワークインターフェース 204 と、メモリ 206 と、これらのコンポーネントを互いに接続するための 1 つ以上の通信バス 208（チップセットと呼ばれる場合もある）とを備える。音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 は、ユーザ入力を容易にする 1 つ以上の入力装置 210 を備える。1 つ以上の入力装置 210 は、オーディオ入力装置 108 または 132（たとえば、音声コマンド入力部またはマイクロホン）を含み、必要に応じて、キーボード、マウス、タッチスクリーンディスプレイ、タッチ入力パッド、ジェスチャーキャプチャカメラ、またはその他の入力ボタンもしくは制御部などのその他の入力装置を含む）。いくつかの実装形態では、音声アシスタントクライアントデバイス 102 は、キーボードを補助または置き換えるために、マイクロホンおよび音声認識、または、カメラおよびジェスチャー認識を使用する。また、音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 は、1 つ以上の出力装置 212 を備える。1 つ以上の出力装置 212 は、オーディオ出力装置 110 または 134（たとえば、1 つ以上のスピーカ、ヘッドフォンなど）を含み、必要に応じて、ユーザインタフェースの提示を可能にし、かつ、コンテンツおよび情報を表示する 1 つ以上の表示装置（たとえば、ディスプレイ 144）および / または 1 つ以上の視覚的インジケータ 142 もしくは 152（たとえば、LED）を含む。必要に応じて、音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 は、音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 の位置を特定するための、GPS（Global Positioning Satellite）または他の地理的位置受信機など、位置検出部 214 を備える。また、音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 は、必要に応じて、他のオブジェクト（たとえば、ウェアラブルパーソナルデバイスの場合、ユーザ / 着用者）への音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 の近接度を判断するための近接検出デバイス 215、たとえば、IR センサを備えてもよい。必要に応じて、音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 は、センサ（複数可）213（たとえば、加速度計、ジャイロスコープなど）を含む。

#### 【0043】

メモリ 206 は、DRAM、SRAM、DDR RAM などの高速ランダムアクセスメモリ、またはその他のランダムアクセス固体記憶装置を含み、必要であれば、1 つ以上の磁気ディスク記憶装置、1 つ以上の光学ディスク記憶装置、1 つ以上のフラッシュメモリデバイスなどの不揮発性メモリ、または 1 つ以上のその他の不揮発性固体記憶装置を含む。メモリ 206 は、必要であれば、1 つ以上の処理装置 202 から離れて位置する 1 つ以

10

20

30

40

50

上の記憶装置を含む。メモリ 206、またはメモリ 206 内の当該不揮発性メモリは、非一時的なコンピュータ読み取り可能な記憶媒体を含む。いくつかの実装形態では、メモリ 206、またはメモリ 206 の非一時的なコンピュータ読み取り可能な記憶媒体は、以下のプログラム、モジュール、およびデータ構造、またはそれらのサブセットもしくはスーパーセットを格納する。

【0044】

様々な基本システムサービス进行处理するための、かつ、ハードウェア依存のタスクを実行するためのプロシージャを含むオペレーティングシステム 216。

【0045】

1つ以上のネットワークインターフェース 204（有線またはワイヤレス）、および、インターネット、その他のワイドエリアネットワーク、ローカルエリアネットワーク、メトロポリタンエリアネットワークなどの1つ以上のネットワーク 112 を介して音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 をその他のデバイス（たとえば、サーバシステム 114、クライアント 102、140、その他の音声アシスタントクライアントデバイス 104 またはキャストデバイス 106）に接続するためのネットワーク通信モジュール 218。

【0046】

1つ以上の出力装置 212（たとえば、ディスプレイ、スピーカなど）を介して音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 で情報の提示を可能にするためのユーザインタフェースモジュール 220。

【0047】

1つ以上のユーザ入力または1つ以上の入力装置 210 がキャプチャまたは受信した対話进行处理し、当該入力および対話を解釈するための入力処理モジュール 222。

【0048】

口頭入力を処理し、当該口頭入力をサーバ 114 に提供し、サーバ 114 から応答を受信し、当該応答を出力するための音声アシスタントモジュール 136。

【0049】

以下を含む、少なくとも音声アシスタントモジュール 136 に関連付けられたデータを格納するためのクライアントデータ 226。

【0050】

音声アシスタントモジュール 136 の設定および構成ならびに音声アシスタント機能に関連付けられた情報を格納するための音声アシスタント設定 228。

【0051】

コンテンツまたは情報の定義済みのおよび/またはユーザ指定のソースおよびカテゴリを格納するためのコンテンツソース/情報ソース 230 およびコンテンツカテゴリ/情報カテゴリ 232。

【0052】

受信したコマンドおよび要求、コマンドおよび要求に対する応答、コマンドおよび要求に応答して行われた動作など、音声アシスタントモジュール 136 の動作および使用に関連付けられた情報（たとえば、ログ）を格納するための利用履歴 234。

【0053】

コンテンツソース/情報ソース 230 におけるユーザのそれぞれのアカウントおよびこれらの認可されたアカウントのアカウント情報にアクセスするための1つ以上のユーザの認可および認証情報を格納するためのユーザアカウントおよび認可 236。

【0054】

コンテンツソースと通信することを含む、キャストデバイス 106 のキャスト機能を実行するための受信モジュール 146。

【0055】

いくつかの実装形態では、音声アシスタントクライアントデバイス 104 またはキャストデバイス 106 は、音声アシスタントおよび関連機能のための1つ以上のライブ

10

20

30

40

50

ライおよび1つ以上のアプリケーションプログラミングインタフェース(API)を含む。これらのライブラリは、音声アシスタントモジュール136または受信モジュール146に含まれてもよく、または、音声アシスタントモジュール136または受信モジュール146によって互いにリンクされてもよい。ライブラリは、音声アシスタント機能または音声アシスタント機能を容易にしたその他の関数に関連付けられたモジュールを含む。APIは、音声アシスタント機能を容易にするハードウェアおよびその他のソフトウェア(たとえば、オペレーティングシステム、その他のアプリケーション)へのインタフェースを提供する。たとえば、音声アシスタントクライアントライブラリ240、デバッグングライブラリ242、プラットフォームAPI244、およびPOSI X API246がメモリ206に格納されてもよい。これらのライブラリおよびAPIについては、図4を参照して、以下にさらに詳しく説明する。

10

#### 【0056】

いくつかの実装形態では、音声アシスタントクライアントデバイス104またはキャストリングデバイス106は、音声アシスタントクライアントライブラリ240のモジュールおよび関数を利用する音声アプリケーション250を含み、必要に応じて、デバッグングライブラリ242、プラットフォームAPI244、およびPOSI X API246を含む。いくつかの実装形態では、音声アプリケーション250は、音声アシスタントクライアントライブラリ240の使用によって音声対応になるファーストパーティまたはサードパーティアプリケーションなどである。

20

#### 【0057】

上記要素の各々は、前述の記憶装置のうちの1つ以上に格納されてもよく、上述の関数を実行するための命令セットに対応する。上記モジュールまたはプログラム(つまり、命令セット)は、別々のソフトウェアプログラム、プロシージャ、モジュール、またはデータ構造として実装される必要がないため、これらのモジュールの様々なサブセットが、様々な実装形態において組み合わせられるまたは並べ替えられてもよい。いくつかの実装形態では、メモリ206は、必要であれば、上記モジュールおよびデータ構造のサブセットを格納する。さらに、メモリ206は、必要であれば、上に記載されないさらなるモジュールおよびデータ構造を格納する。

#### 【0058】

図3は、いくつかの実装形態に係る、ネットワーク環境100のサーバシステム114の例を示すブロック図である。サーバ114は、通常、1つ以上の処理装置(CPU)302と、1つ以上のネットワークインターフェース304と、メモリ306と、これらのコンポーネント(チップセットと呼ばれる場合もある)を互いに接続するための1つ以上の通信バス308とを備える。サーバ114は、必要に応じて、キーボード、マウス、音声コマンド入力部またはマイクロホン、タッチスクリーンディスプレイ、タッチ入力パッド、ジェスチャーキャプチャカメラ、またはその他の入力ボタンもしくは制御部など、ユーザ入力を容易にする1つ以上の入力装置310を備える。さらには、サーバ114は、キーボードを補助または置き換えるために、マイクロホンおよび音声認識、または、カメラおよびジェスチャー認識を使用してもよい。いくつかの実装形態では、サーバ114は、必要に応じて、たとえば、電子デバイス上にプリントされた図形シリーズコードを撮影するための1つ以上のカメラ、スキャナ、または光センサ部を備える。また、サーバ114は、必要に応じて、ユーザインタフェースの提示を可能にし、かつ、コンテンツを表示する、1つ以上のスピーカおよび/または1つ以上の表示装置を含む、1つ以上の出力装置312を備える。

30

40

#### 【0059】

メモリ306は、DRAM、SRAM、DDR RAMなどの高速ランダムアクセスメモリ、またはその他のランダムアクセス固体記憶装置を含み、必要であれば、1つ以上の磁気ディスク記憶装置、1つ以上の光学ディスク記憶装置、1つ以上のフラッシュメモリデバイスなどの不揮発性メモリ、または1つ以上のその他の不揮発性固体記憶装置を含む。メモリ306は、必要であれば、1つ以上の処理装置302から離れて位置する1つ以

50

上の記憶装置を含む。メモリ 306、またはメモリ 306 内の当該不揮発性メモリは、非一時的なコンピュータ読み取り可能な記憶媒体を含む。いくつかの実装形態では、メモリ 306、またはメモリ 306 の非一時的なコンピュータ読み取り可能な記憶媒体は、以下のプログラム、モジュール、およびデータ構造、またはそれらのサブセットもしくはスーパーセットを格納する。

【0060】

様々な基本システムサービス进行处理するための、かつ、ハードウェア依存のタスクを実行するためのプロシージャを含むオペレーティングシステム 316。

【0061】

1つ以上の処理装置 304（有線またはワイヤレス）、および、インターネット、その他のワイドエリアネットワーク、ローカルエリアネットワーク、メトロポリタンエリアネットワークなどの1つ以上のネットワーク 112を介してサーバシステム 114をその他のデバイス（たとえば、音声アシスタントクライアントデバイス 104、キャストイングデバイス 106、クライアント 102、クライアント 140）に接続するためのネットワーク通信モジュール 318。

【0062】

クライアントデバイス 104またはキャストイングデバイス 106の位置情報に基づいて、音声アシスタントクライアントデバイス 104またはキャストイングデバイス 106の近接度および/または位置を特定するための近接度/位置特定モジュール 320。

【0063】

少なくとも以下のうちの1つ以上を含む、音声アシスタントの口頭入力（たとえば、音声アシスタントクライアントデバイス 104およびキャストイングデバイス 106から受信した口頭入力）进行处理するための音声アシスタントバックエンド 116。

【0064】

口頭入力进行处理し、口頭入力に含まれるコマンドおよび要求を識別するための口頭入力処理モジュール 324。

【0065】

コマンドおよび要求に対するコンテンツおよび情報応答を集めるためのコンテンツ/情報収集モジュール 326。

【0066】

コマンドおよび要求に応答する音声出力を生成し、応答であるコンテンツおよび情報とともに当該音声出力を追加するための応答生成モジュール 328。

【0067】

以下を含む、少なくとも音声アシスタントプラットフォームの動作に関連付けられたデータを格納するサーバシステムデータ 330。

【0068】

以下を含む、音声アシスタントプラットフォームのユーザに関連付けられた情報を格納するためのユーザデータ 332。

【0069】

- 音声アシスタント設定 228に対応する音声アシスタント設定情報と、コンテンツソース/情報ソース 230およびコンテンツカテゴリ/情報カテゴリ 232に対応する情報を格納するためのユーザ音声アシスタント設定 334。

【0070】

- コマンドおよび要求の履歴、ならびに対応する応答を含む、音声アシスタントについてのユーザの履歴（たとえば、ログ）を格納するためのユーザ履歴 336。

【0071】

- コンテンツソース/情報ソース 230におけるユーザのそれぞれのアカウントにアクセスするためのユーザの認可および認証情報と、ユーザアカウントおよび認可 236に対応する、これらの認可されたアカウントのアカウント情報とを格納するためのユーザアカウントおよび認可 338。

10

20

30

40

50

## 【 0 0 7 2 】

上記要素の各々は、前述の記憶装置のうちの1つ以上に格納されてもよく、上述の関数を実行するための命令セットに対応する。上記モジュールまたはプログラム（つまり、命令セット）は、別々のソフトウェアプログラム、プロシージャ、モジュール、またはデータ構造として実装される必要がないため、これらのモジュールの様々なサブセットが、様々な実装形態において組み合わせられるまたは並べ替えられてもよい。いくつかの実装形態では、メモリ206は、必要であれば、上記モジュールおよびデータ構造のサブセットを格納する。さらに、メモリ206は、必要であれば、上に記載されないさらなるモジュールおよびデータ構造を格納する。

## 【 0 0 7 3 】

いくつかの実装形態では、音声アシスタントモジュール136（図2）は、1つ以上のライブラリを含む。ライブラリは、それぞれの関数を実行するモジュールまたはサブモジュールを含む。たとえば、音声アシスタントクライアントライブラリは、音声アシスタントの関数を実行するモジュールを含む。また、音声アシスタントモジュール136は、特定のハードウェア（たとえば、クライアントデバイスまたはキャストリングデバイス上のハードウェア）、特定のオペレーティングソフトウェア、または遠隔システムと協働するための1つ以上のアプリケーションプログラミングインタフェース（API）を含んでもよい。

## 【 0 0 7 4 】

いくつかの実装形態では、ライブラリは、たとえば、帯域通過処理、フィルタリング処理、消去処理、およびホットワード検出を含む、オーディオ信号処理動作をサポートするモジュールを含む。いくつかの実装形態では、ライブラリは、バックエンド（たとえば、サーバベースの）音声処理システムに接続するためのモジュールを含む。いくつかの実装形態では、ライブラリは、デバッグ（たとえば、音声認識のデバッグ、ハードウェア問題のデバッグ、自動テスト）用のモジュールを含む。

## 【 0 0 7 5 】

図4は、音声アシスタントクライアントデバイス104またはキャストリングデバイス106に格納され得、かつ、音声アシスタントモジュール136または別のアプリケーションによって実行され得るライブラリおよびAPIを示す図である。ライブラリおよびAPIは、音声アシスタントクライアントライブラリ240と、デバッグライブラリ242と、プラットフォームAPI244と、POSIX API246とを含んでもよい。音声アシスタントクライアントデバイス104またはキャストリングデバイス106におけるアプリケーション（たとえば、音声アシスタントモジュール136、音声アシスタントとの協働をサポートしたいであろうその他のアプリケーション）は、当該アプリケーションにおいて音声アシスタント機能を提供またはサポートするために、これらのライブラリおよびAPIを含んでもよく、またはそれらにリンクされてもよく、かつ、当該ライブラリおよびAPIを実行してもよい。いくつかの実装形態では、音声アシスタントクライアントライブラリ240とデバッグライブラリ242は別個のライブラリである。音声アシスタントクライアントライブラリ240とデバッグライブラリ242とを別々に分けておくことによって、これらのライブラリのセキュリティ上の異なる影響を考慮に入れた異なるリリースおよび更新プロシージャが容易になる。

## 【 0 0 7 6 】

いくつかの実装形態では、これらのライブラリは、柔軟性がある。ライブラリは、複数のデバイスタイプ間で使用してもよく、同じ音声アシスタント機能を組み込んでもよい。

## 【 0 0 7 7 】

いくつかの実装形態では、ライブラリは、標準の共用オブジェクト（たとえば、標準のLinux（登録商標）共用オブジェクト）に依存するため、これらの標準の共用オブジェクトを利用する異なるオペレーティングシステムまたはプラットフォームと互換性がある（たとえば、組込み用Linuxの様々なLinuxディストリビューションおよびフレーバー）。

10

20

30

40

50

## 【 0 0 7 8 】

いくつかの実装形態では、P O S I X A P I 2 4 6 は、様々なオペレーティングシステムとの互換性のために、標準 A P I を提供する。よって、音声アシスタントクライアントライブラリ 2 4 0 は、P O S I X に準拠する異なるオペレーティングシステムのデバイスに含まれてもよく、P O S I X A P I 2 4 6 は、音声アシスタントクライアントライブラリ 2 4 0 と異なるオペレーティングシステムとの間の互換インタフェースを提供する。

## 【 0 0 7 9 】

いくつかの実装形態では、ライブラリは、音声アシスタントを実装する異なる種類のデバイス（たとえば、タイマ、アラーム、音量調節）間で利用可能なベースユースケースをサポートおよび容易にするためのモジュールを含む。

10

## 【 0 0 8 0 】

いくつかの実装形態では、音声アシスタントクライアントライブラリ 2 4 0 は、音声アシスタントを起動、設定、および音声アシスタントと対話するための関数またはモジュールを含んだコントローライントラフェース 4 0 2 を含む。いくつかの実装形態では、コントローライントラフェース 4 0 2 は、デバイスにおいて音声アシスタントを開始するための「S t a r t ( )」関数またはモジュール 4 0 4 と、音声アシスタントにアクションを登録する（たとえば、音声アシスタントを介してアクションが実施可能になり得るように）ための「R e g i s t e r A c t i o n ( )」関数またはモジュール 4 0 6 と、更新された設定を用いて音声アシスタントの再設定するための「R e c o n f i g u r e ( )」4 0 8 関数と、基本イベント用の関数のセットをアシスタントに登録するための「R e g i s t e r E v e n t O b s e r v e r ( )」関数 4 1 0 とを含む。

20

## 【 0 0 8 1 】

いくつかの実装形態では、音声アシスタントクライアントライブラリ 2 4 0 は、特定の音声アシスタント機能に関連付けられた複数の関数またはモジュールを含む。たとえば、ホットワード検出モジュール 4 1 2 は、音声入力を処理してホットワードを検出する。音声処理モジュール 4 1 4 は、音声入力に含まれる音声进行处理し、音声をテキストに変換する、または、テキストを音声に変換する（たとえば、単語および表現の識別、音声からテキストデータへの変換、テキストデータから音声への変換）。アクション処理モジュール 4 1 6 は、口頭入力に回答したアクションおよび動作を行う。ローカルタイマ / アラーム / 音量調節モジュール 4 1 8 は、デバイスにおける目覚まし時計、タイマ、および音量調節機能、ならびに音声入力によるそれらの制御を容易にする（たとえば、デバイスにおけるタイマ、時計、目覚まし時計を管理する）。ロギング / 評価指標モジュール 4 2 0 は、音声入力および応答を記録し（たとえば、ログを取る）、関連する評価指標（たとえば、応答時間、アイドル時間など）を判断および記録する。オーディオ入力処理モジュール 4 2 2 は、音声入力のオーディオ进行处理する。M P 3 デコーディングモジュール 4 2 4 は、M P 3 にエンコードされたオーディオをデコードする。オーディオ入力モジュール 4 2 6 は、オーディオ入力装置（たとえば、マイクロホン）からオーディオをキャプチャする。オーディオ出力モジュール 4 2 8 は、オーディオ出力装置（たとえば、スピーカ）からオーディオを出力する。デバイスにおける音声アシスタントに関連付けられたイベントをキューに入れる、かつ、デバイスにおける音声アシスタントの状態を追跡するためのイベントキューイング / 状態追跡モジュール 4 3 0 。

30

40

## 【 0 0 8 2 】

いくつかの実装形態では、デバッグライブラリ 2 4 2 は、デバッグ用のモジュールおよび関数を提供する。たとえば、H T T P サーバモジュール 4 3 2 は、接続性問題のデバッグを容易にし、デバッグサーバ / オーディオストリーミングモジュール 4 3 4 は、オーディオ問題をデバッグする。

## 【 0 0 8 3 】

いくつかの実装形態では、プラットフォーム A P I 2 4 4 は、音声アシスタントクライアントライブラリ 2 4 0 とデバイスのハードウェア機能との間のインタフェースを提供す

50



る。たとえば、プラットフォーム A P I は、デバイスに対するボタン入力をキャプチャするためのボタン入力インタフェース 4 3 6 と、ループバックオーディオをキャプチャするためのループバックオーディオインタフェース 4 3 8 と、評価指標のログを取る、かつ、判断するためのロギング / 評価指標インタフェース 4 4 0 と、オーディオ入力をキャプチャするためのオーディオ入力インタフェース 4 4 2 と、オーディオを出力するためのオーディオ出力インタフェース 4 4 4 と、音声アシスタントと対話し得るその他のサービスを用いてユーザを認証するための認証インタフェース 4 4 6 とを含む。図 4 に示す音声アシスタントクライアントライブラリ編成の利点は、一貫した A P I と、音声アシスタントの関数のセットとを有する様々な音声アシスタントのデバイスタイプ上で、同じまたは同様の音声処理機能を提供できることである。この一貫性は、音声アシスタントアプリケーションのポータビリティと、音声アシスタント動作の一貫性とをサポートして、一貫したユーザ・インタラクションならびに異なるデバイスタイプ上で動作する音声アシスタントアプリケーションおよび関数に詳しくなることを促進する。いくつかの実装形態では、音声アシスタントクライアントライブラリ 2 4 0 のすべてまたは一部は、サーバベースの音声アシスタントアプリケーション（たとえば、処理するためにサーバ 1 1 4 に送信された音声入力に対して動作サーバアプリケーション）をサポートするために、サーバ 1 1 4 において提供されてもよい。

10

**【 0 0 8 4 】**

コントローラ 4 0 2 (「C o n t r o l l e r」) に対応するクラスおよび関数ならびに関連クラスのコード例を、以下に示す。これらのクラスおよび関数は、共通 A P I を介して、いろいろなデバイス上で実行可能なアプリケーションによって採用され得る。

20

**【 0 0 8 5 】**

下記クラス「A c t i o n M o d u l e」は、音声アシスタントサーバが提供するコマンドを処理するためにアプリケーションが当該アプリケーションのモジュールを登録することを容易にする。

**【 0 0 8 6 】**

## 【数 1】

//アプリケーションは、音声アシスタントサーバが提供するコマンドを処理するために  
//当該アプリケーションのソフトウェアモジュールを登録できる。

```
class ActionModule {
    public:
//アクション結果は、アクションが無事実行されたかどうかを記述する。
        class Result {
            public:
                virtual ~Result() = default;
//成功を示すようにアクション結果を設定する。
                virtual void SetOk() = 0;
//所与の応答コードおよび人間が理解できる文字列にアクション結果を設定する。
                virtual void SetError(int response_code,
                    const std::string& str);
        };
//アクションハンドラへの引き数。
        class Args {
            public:
                virtual ~Args() = 0;
//所与の型のアクションハンドラの引き数についてのシリアル化された
//protobuf データをゲット。
                virtual bool GetProtobufDataFromType
                    (std::string type, std::string*
                    data) = 0;
        };
        virtual ~ActionModule() = 0;
//このモジュールの名前を返す。
        virtual std::string GetName() = 0;
//所与の|action_name|をその|args|を用いて処理し、アクションの実行の成果に応じて
//結果を更新する。
        virtual void Handle(std::string action_name,
            std::unique_ptr<Args> args, Result* result) = 0;
//所与のシリアル化データに指定されたprotobufを設定し、このモジュールの
//ローカル状態を音声アシスタントに示す。
        virtual bool GetModuleContext(std::string* protobuf_type,
            std::string* protobuf_data) = 0;
};
```

## 【0087】

下記クラス「BuildInfo」を用いて、音声アシスタントクライアントライブラリ240を実行中のアプリケーションまたは音声アシスタントクライアントデバイス104自体を記述してもよい（たとえば、アプリケーション、プラットフォーム、および/またはデバイスの識別子またはバージョン番号を用いて）。

## 【0088】

## 【 数 2 】

```
//音声アシスタントクライアントライブラリを実行中のアプリケーションを記述する
//ために使用されるビルド情報。
//専用の音声アシスタントデバイスについては、これは、デバイスを記述する。

//このオブジェクトは、CreateDefaultBuildInfo から返され、
//変更され得、その後、Settings オブジェクトに戻される。

class BuildInfo {
    public:
        virtual ~BuildInfo() = default;
//アプリケーションのバージョンを設定する。
        virtual void SetApplicationVersion(const std::string&
            application_version) = 0;
//インストール識別子を設定する。これは、その他のデバイスまたはユーザ識別子と
//同じではないデバイス固有の識別子でなければならない。
        virtual void SetInstallId(const std::string&
            install_id) = 0;
//プラットフォーム識別子を設定する。
        virtual void SetPlatformId(const std::string&
            platform_id) = 0;
//プラットフォームのバージョンを設定する。
        virtual void SetPlatformVersion(const std::string&
            platform_version) = 0;
//デバイスの機種を設定する。任意。
        virtual void SetDeviceModel(const std::string&
            device_model) = 0;
};
```

10

20

## 【 0 0 8 9 】

30

下記クラス「EventDelegate」は、音声認識の開始、音声アシスタントの  
応答の出力の開始および完了など、基本イベントに関連付けられた関数を定義する。

## 【 0 0 9 0 】

## 【 数 3 】

//アシスタントライブラリからイベントを受け取る。

```
class EventDelegate {  
    public:  
        class RecognizedSpeechChangedEvent {  
            public:  
                virtual ~RecognizedSpeechChangedEvent() {}  
  
                virtual std::string  
                    GetRecognizedSpeech() = 0;  
        };  
        virtual ~EventDelegate() {}  
  
        //音声アシスタントが認識したテキストの更新を示す。  
        // OnRecognizedSpeechFinishedEvent の一部である場合、これは、最終的に認識された  
        // テキストを示す。  
  
        virtual void OnBootingUpEvent() = 0;  
  
        //音声アシスタントクライアントライブラリが起動中であることを示す。  
  
        virtual void OnHeardHotwordEvent() = 0;  
  
        //音声の認識が開始したことを示す。音声認識は、OnRecognizingSpeechFinishedEvent が  
        //受け付けられるまで続く。  
  
        virtual void OnRecognizingSpeechStartedEvent() = 0;  
  
        //認識された音声の現在の仮説が変更されたことを示す。  
        //|event|は、新しい仮説を示す。  
  
        virtual void OnRecognizedSpeechChangedEvent(  
            const RecognizedSpeechChangedEvent& event) = 0;  
  
        //最後の音声認識が発生したことを示す。  
        //|event|は、最終値を示す。  
  
        virtual void OnRecognizingSpeechFinishedEvent(  
            const RecognizedSpeechChangedEvent& event) = 0;  
  
        //音声アシスタントが音声による応答を開始したことを示す。  
        //音声アシスタントは、OnRespondingFinishedEvent が受け付けられるまで応答  
        //し続ける。  
  
        virtual void OnRespondingStartedEvent() = 0;
```

10

20

30

//音声アシスタントが音声による応答を終了したことを示す。

```
virtual void OnRespondingFinishedEvent() = 0;
```

//アラームが鳴り始めたことを示す。アラームは、OnAlarmSoundingFinishedEvent が  
//受け付けられるまで鳴り続ける。

```
virtual void OnAlarmSoundingStartedEvent() = 0;
```

//アラームが鳴り終わったことを示す。

```
virtual void OnAlarmSoundingFinishedEvent() = 0;
```

//タイマが鳴り始めたことを示す。タイマは、OnTimerSoundingFinishedEvent が  
//受け付けられるまで鳴り続ける。

```
virtual void OnTimerSoundingStartedEvent() = 0;
```

//タイマが鳴り終わったことを示す。

```
virtual void OnTimerSoundingFinishedEvent() = 0;
```

//デフォルト音量に対して音量の変更があったことを示す（これは、ユーザが、アラーム  
//または別の特定の音量の種類を指定することなく、たとえば、「音量を上げて」  
//といったときに発生する。）|new\_volume|は、0.0 から 1.0 までの  
//新しいデフォルト音量を示す。

```
virtual void OnDefaultVolumeChangeEvent(float  
new_volume) = 0;
```

//音声アシスタントクライアントライブラリのサーバとの有効期限が切れており、  
//更新する必要があることを示す。これが発生したとき、クライアントは、  
//サーバと対話できなくなる。

```
virtual void OnClientLibraryOutOfDateEvent() = 0;
```

```
};
```

【 0 0 9 1 】

下記クラス「DefaultEventDelegate」は、特定のイベントについて、何もしないオーバーライドの関数を定義する。

【 0 0 9 2 】

10

20

30

## 【 数 4 】

// EventDelegate のデフォルトの何もしない実装を提供する。  
//対象の関数のみをオーバーライドするのに有効である。

```
class DefaultEventDelegate : public EventDelegate {  
    public:  
        void OnBootingUpEvent() override {}  
        void OnHeardHotwordEvent() override {}  
        void OnRecognizingSpeechStartedEvent() override {}  
        void OnRecognizedSpeechChangedEvent(const  
            RecognizedSpeechChangedEvent& event) override {}  
        void OnRecognizingSpeechFinishedEvent(const  
            RecognizedSpeechChangedEvent& event) override {}  
        void OnRespondingStartedEvent() override {}  
        void OnRespondingFinishedEvent() override {}  
        void OnAlarmSoundingStartedEvent() override {}  
        void OnAlarmSoundingFinishedEvent() override {}  
        void OnTimerSoundingStartedEvent() override {}  
        void OnTimerSoundingFinishedEvent() override {}  
        void OnDefaultVolumeChangeEvent(float new_volume)  
            override {}  
        void OnClientLibraryOutOfDateEvent() override {}  
};
```

10

20

## 【 0 0 9 3 】

下記クラス「Settings」は、コントローラ402に提供され得る設定（たとえば、ロケール、地理的位置、ファイルシステムのディレクトリ）を定義する。

## 【 0 0 9 4 】

## 【 数 5 】

//コントローラに提供されるアシスタント設定。アシスタントを開始するときに  
 //コントローラに提供されなければならない。また、反映されるために、更新されてから  
 //Reconfigure 関数に与えられ得る。  
 //組み込みアプリは、これから派生するアプリ独自のクラスを作成してはいけない。

```

class Settings {
    public:
        virtual ~Settings() {}

//デフォルトの BuildInfo オブジェクトを作成する。
        virtual std::unique_ptr<BuildInfo>
            CreateDefaultBuildInfo() = 0;

//デバイスの地理的位置を設定する。任意。
        virtual void SetGeolocation(const Geolocation&
            geolocation) = 0;

//デバイスのビルド情報を設定する。
        virtual void SetBuildInfo(const BuildInfo&
            build_info) = 0;

//音声アシスタントクライアントライブラリが利用できるファイルシステムの
//ディレクトリを設定する。このディレクトリは、出荷時の設定にリセットされたとき
//など、音声アシスタントクライアントライブラリが前のコンテキストのすべてを
//失う度にクリアされる。
        virtual void SetAssistantDirectory(const
            std::string& path) = 0;

//接続をサーバに渡すための UserAgent を設定する。
        virtual void SetUserAgent(const std::string&
            user_agent) = 0;

//デバイスのロケールを設定する。
        virtual void SetLocaleInfo(const LocaleInfo&
            locale_info) = 0;

};

```

## 【 0 0 9 5 】

下記クラス「Controller」は、コントローラ402に対応し、Start( )、Reconfigure( )、RegisterAction( )、およびRegisterEventObserver( )関数は、関数Start( )404、Reconfigure( )408、RegisterAction( )406、およびRegisterEventObserver( )410にそれぞれ対応する。

## 【 0 0 9 6 】

40

## 【数 6】

```

//アシスタントの Controller クラス
class Controller {
    public:
        virtual ~Controller() {}

//アプリケーションが構成してから Start に渡す新しいデフォルト設定オブジェクトを
//作成する。

        virtual std::unique_ptr<Settings>
            CreateDefaultSettings() = 0;

//アシスタントを開始し、すぐに返す。成功の場合、true を返し、失敗の場合、false を返す。
//プロセスにつき1回しか継承しない。|settings|は、Assistant モジュールの設定である。
//const 参照によって渡されるので、呼び出し側が Settings オブジェクトを保存することと、
//それに続くいずれの変更も、Reconfigure に渡されない限り影響がないことが明らかである。
//この関数は、必要な設定が行われていない場合、失敗する。

        virtual bool Start(const Settings& settings) = 0;

//動作中のアシスタントを再設定し、すぐに返す。
//アシスタントがまだ開始していない場合など、失敗の場合、false を返す。
//|settings|は、音声アシスタントモジュールの新しい設定である。この関数は、
//必要な設定が行われていない場合、失敗する。

        virtual bool Reconfigure(const Settings& settings) = 0;

//アクション|module|を登録する。すでに登録済みの場合、失敗する。

        virtual bool RegisterAction(std::unique_ptr<ActionModule>
            module) = 0;

//すべてのアシスタントイベントを受け付けるために、EventDelegate を登録する。

        virtual void RegisterEventObserver(
            std::unique_ptr<EventDelegate> delegate) = 0;

//この関数を呼び出して、アシスタントを制御するコントローラクラスを作成する。
//|platform|は、アシスタントが使用するプラットフォームAPIへのポインタに
//設定されなければならない。エラーが発生した場合は、nullptr を返す。

        static ASSISTANT_EXPORT std::unique_ptr<Controller>
            Create(std::unique_ptr<PlatformApi> platform_api);
};

```

## 【0097】

いくつかの実装形態では、音声アシスタントクライアントデバイス104またはキャストデバイス106は、プラットフォームを実装する（たとえば、同じプラットフォームを利用する他のデバイスと通信するためのインタフェースのセット、および当該インタフェースのセットをサポートするように構成されたオペレーティングシステム）。下記のコード例は、音声アシスタントクライアントライブラリ402が当該プラットフォームと対話するためのインタフェースに関連付けられた関数を示す。

## 【0098】

下記クラス「Authentication」は、特定のアカウントを有する音声アシスタントのユーザを認証するための認証トークンを定義する。

## 【0099】

10

20

30

40



## 【数 7】

//プラットフォームの認証プロバイダ。

```
class Authentication {
    public:
```

//認証トークンの認証範囲を返す。

```
virtual std::string GetGoogleOAuth2Scopes() = 0;
```

//認証トークンを返す。

```
virtual bool GetGoogleOAuth2Token(std::string*
    token) = 0;
```

10

```
protected:
```

```
virtual ~Authentication() = default;
```

```
};
```

## 【0100】

下記クラス「OutputStreamType」は、オーディオ出力ストリームの型を定義する。

## 【0101】

## 【数 8】

20

//可能なオーディオ出力ストリームの型。

```
enum class OutputStreamType {
    kTts,
    kAlarm,
    kCalibration,
};
```

## 【0102】

下記クラス「SampleFormat」は、サポートするオーディオサンプルのフォーマット（たとえば、PCMフォーマット）を定義する。

30

## 【0103】

## 【数 9】

//サポートするPCMサンプルのフォーマット。

```
enum class SampleFormat {
    kInterleavedS16,    //インターリーブされた符号付き16ビットのInteger型。
    kInterleavedS32,    //インターリーブされた符号付き32ビットのInteger型。
    kInterleavedF32,    //インターリーブされた32ビットのfloat型。
    kPlanarS16,         //プレーナー（planar）符号付き16ビットのInteger型。
    kPlanarS32,         //プレーナー符号付き32ビットのInteger型。
    kPlanarF32,         //プレーナー32ビットのfloat型。
};
```

40

## 【0104】

下記「BufferFormat」は、デバイスのオーディオバッファに格納されるデータのフォーマットを定義する。

## 【0105】

## 【数 1 0】

//オーディオバッファに格納されるデータのフォーマットについての情報。

```
struct BufferFormat {  
    int sample_rate;  
    SampleFormat sample_format;  
    int num_channels;  
};
```

## 【0 1 0 6】

10

下記クラス「AudioBuffer」は、オーディオデータのバッファを定義する。

## 【0 1 0 7】

## 【数 1 1】

//入／出力オーディオデータのバッファクラス。

```
class AudioBuffer {  
    public:
```

//バッファに含まれるデータのフォーマットを返す。

```
    virtual BufferFormat GetFormat() const = 0;
```

20

//不変データ。受信データを読み出すためにAudioInput デリゲートによって  
//使用される。

```
    virtual const char* GetData() const = 0;
```

//書き込み可能なデータ。出力するためのデータをさらに書き出すためにAudioOutput  
//デリゲートによって使用される。

```
    virtual char* GetWritableData() const = 0;
```

// GetData()/GetWritableData()に含まれるオーディオフレームの数を返す。

```
    virtual int GetFrames() const = 0;
```

```
    protected:
```

30

```
    virtual ~AudioBuffer() {}
```

```
};
```

## 【0 1 0 8】

下記クラス「AudioOutput」は、オーディオ出力用のインタフェースを定義する。

## 【0 1 0 9】

## 【数 1 2】

//オーディオ出力用のインタフェース。

```

class AudioOutput {
public:
    enum Error {
        kFatalError,
        kUnderrun,
    };
    class Delegate {
public:
//出力オーディオデータがさらに必要な場合に呼び出される。デリゲートの実装は、
//|buffer|にデータをなるべく早く充填しなければならず、
//なんらかのデータが書き込まれると、|done_cb|を1回呼び出す。なお、デリゲートは、
//このバッファを部分的に充填してもよいが、|bytes_written|の数は、フレームサイズの
//倍数でなければならない。デリゲートは、|buffer|を所有しない。
//なお、このメソッドはブロックしない。
//バッファを速やかに充填するためのデータがない場合、バッファは、いずれかの
//スレッドによって非同期で充填され得、その後、|done_cb|が呼び出されなければならない。
//|done_cb|は、Stop()の呼び出しによってストリームが停止した後に呼び出されては
//ならない。ストリームの終端に到達した場合、デリゲートは、0 |bytes_written|を
//用いて|done_cb|を呼び出さなければならない。
        virtual void FillBuffer(AudioBuffer* buffer,
                                const std::function<void(int
                                    frames_written)>& done_cb) = 0;
//ストリームの終端（すなわち、デリゲートが0 |bytes_writted|をFillBuffer()の
//|done_cb|に渡したポイント）が再生されたことを示すために呼び出される。
//これが呼び出されると、再生されなかったデータを破棄してしまう恐れなしに
//Stop()を安全に呼び出すことができる。
        virtual void OnEndOfStream() = 0;
//出力エラーが発生したときに呼び出される。
        virtual void OnError(Error error) = 0;
//出力が停止されると呼び出される。このメソッドが呼び出されると、出力が再び開始
//されない限り、デリゲートメソッドは呼び出されない。
        virtual void OnStopped() = 0;
    protected:
        ~Delegate() {}
    };
    virtual ~AudioOutput() {}

```

//この出力のストリーム型を返す。ストリーム型は、当該出力が作成されたときに  
//指定される。

```
virtual OutputStreamType GetType() = 0;
```

//オーディオ出力を開始する。これは、|delegate|の FillBuffer() メソッドを  
//呼び出すことによって、所与の|format|のバッファの要求を開始する。

```
virtual void Start(const BufferFormat& format,  
                  Delegate* delegate) = 0;
```

//オーディオ出力を停止し、このインタフェースを、新しいオーディオフォーマット  
//およびデリゲートを用いて Start() を再び安全に呼び出すことができる状態にする。  
//デリゲートが提供したデータで再生されなかったものは、Stop() が呼び出された  
//ときに破棄される。

10

//停止が完了し、デリゲートがこれ以上呼び出しされないと、デリゲートの  
//OnStopped() メソッドが呼び出される。

```
virtual void Stop() = 0;
```

//この出力ストリームの音量範囲を設定する。このストリームの音量は、デフォルト音量が、  
//|min\_volume| <= volume <= |max\_volume| 範囲であるうちは、その音量をたどる  
//（そのため、デフォルト音量を使用するが、所与の範囲に留められる）。  
//|min\_volume|および|max\_volume|は、 $0.0 \leq v \leq 1.0$  の値であり、システムの  
//可能な総出力音量に対する割合を表す。

20

```
virtual void SetVolume(float min_volume, float  
                      max_volume) = 0;
```

```
};
```

【 0 1 1 0 】

下記クラス「AudioInput」は、オーディオ入力をキャプチャするためのインタフェースを定義する。

【 0 1 1 1 】

## 【数 1 3】

//オーディオ入力をキャプチャするためのインタフェース。開始されると、すべての  
 //マイクロホンからオーディオをキャプチャし、各マイクロホンからのデータを、  
 //デリゲートの OnBufferAvailable() メソッドに提供されるバッファにおける  
 //別々のチャンネルとして提供する。

```
class AudioInput {
    public:
        enum Error {
            kFatalError,
            kOverrun,
        };
        class Delegate {
            public:
                //さらに多くの入力オーディオデータが利用可能な場合に呼び出される。
                //|timestamp|は、|buffer|のデータがキャプチャされたときの (CLOCK_MONOTONIC_RAW 時
                //間に対する) マイクロ秒単位の時間である (ループバックオーディオについては、
                //データが再生されると予想された時間を示すタイムスタンプである)。
                virtual void OnBufferAvailable(
                    const AudioBuffer& buffer,
                    int64_t timestamp) = 0;
                // AudioInput 上でエラーが発生すると呼び出される。
                virtual void OnError(Error error) = 0;
                //入力が停止されると呼び出される。このメソッドが呼び出されると、入力がやり直され
                //ない限り、デリゲートメソッドは呼び出されない。
                virtual void OnStopped() = 0;
        };
        virtual ~AudioInput() {}
        //オーディオ入力のキャプチャを開始して|delegate|の OnBufferAvailable() メソッド
        //に渡す。
        virtual void Start(Delegate* delegate) = 0;
        //オーディオ入力のキャプチャを停止する。入力が停止し、デリゲートメソッドの
        //呼び出しがされなくなると、デリゲートの OnStopped() メソッドが呼び出される。
        virtual void Stop() = 0;
};
```

## 【0 1 1 2】

下記クラス「Resources」は、システムリソースへのアクセスを定義する。

## 【0 1 1 3】

10

20

30

40

## 【数 1 4】

//システムリソースファイルへのアクセス。

```
class Resources {  
    public:  
        using ResourceLoadingCallback = std::function<void(  
            const std::string& output)>;  
        Resources() {}  
        virtual ~Resources() {}  
        virtual bool GetBuiltinHotwordData(  
            const LocaleInfo* locale,  
            const ResourceLoadingCallback& callback) = 0;  
        virtual bool GetAlarmMp3(const ResourceLoadingCallback&  
            callback) = 0;  
        virtual bool GetTimerMp3(const ResourceLoadingCallback&  
            callback) = 0;  
        virtual bool GetCalibrationMp3(const  
            ResourceLoadingCallback& callback) = 0;  
        virtual bool GetVolumeChangeMp3(const  
            ResourceLoadingCallback& callback) = 0;  
        virtual bool GetSpeechRecognitionErrorMp3(  
            const LocaleInfo* locale,  
            const ResourceLoadingCallback& callback) = 0;  
        virtual bool GetSpeechRecognitionStoppedMp3(  
            const LocaleInfo* locale,  
            const ResourceLoadingCallback& callback) = 0;  
        virtual bool GetNoInternetMp3(const LocaleInfo* locale,  
            const ResourceLoadingCallback& callback) = 0;  
};
```

10

20

## 【0 1 1 4】

30

下記クラス「PlatformApi」は、音声アシスタントクライアントライブラリ 240 用のプラットフォーム API を指定する（たとえば、プラットフォーム API 244）。

## 【0 1 1 5】

## 【数 1 5】

//音声アシスタントに使用されるプラットフォームAPI。

```
class PlatformApi {
public:
    virtual ~PlatformApi() {}

//所望の|type|のストリーム用のオーディオ出力インタフェースを返す。
//これは、PlatformApi が所有する。
    virtual std::unique_ptr<AudioOutput> GetAudioOutput(
        OutputStreamType type) = 0;
//オーディオ入力をキャプチャするためのインタフェースを返す。
    virtual std::unique_ptr<AudioInput> GetAudioInput() = 0;
//ループバックオーディオをキャプチャするためのインタフェースを返す。これは、
//キャプチャされたデータが、再生されることになっていたオーディオデータ
//である、「オーディオ入力」である。
//ループバックオーディオは、すべてのミキシングおよび後処理が済んだ後、
//出力ハードウェアに送信する前に、できるだけ早くキャプチャされてもよい。
    virtual std::unique_ptr<AudioInput>
        GetLoopbackInput() = 0;
    virtual Authentication& GetAuthentication() = 0;
};
```

10

20

## 【0 1 1 6】

いくつかの実装形態では、音量調節は、音声アシスタントクライアントライブラリ 2 4 0 の外部で処理されてもよい。たとえば、システム音量は、音声アシスタントクライアントライブラリ 2 4 0 によって制御されないデバイスによって管理されてもよい。別の例として、音声アシスタントクライアントライブラリ 2 4 0 は、音量調節を引き続きサポートしてもよいが、音声アシスタントクライアントライブラリ 2 4 0 に対する音量調節についての要求は、デバイスに向けられる。

30

## 【0 1 1 7】

いくつかの実装形態では、音声アシスタントクライアントライブラリ 2 4 0 に含まれるアラームおよびタイマ機能は、ユーザによって無効にされてもよく、または、デバイスにおいてライブラリを実装するときに無効にされてもよい。

## 【0 1 1 8】

また、いくつかの実装形態では、音声アシスタントクライアントライブラリ 2 4 0 は、デバイス上のLEDへのインタフェースをサポートし、デバイスのLEDのLEDアニメーションの表示を容易にする。

## 【0 1 1 9】

いくつかの実装形態では、音声アシスタントクライアントライブラリ 2 4 0 は、キャストデバイス 1 0 6 におけるキャスト受信モジュール（たとえば、受信モジュール 1 4 6）に含まれるまたはそれにリンクされてもよい。音声アシスタントクライアントライブラリ 2 4 0 と受信モジュール 1 4 6 とのリンクは、たとえば、さらなるアクション（たとえば、ローカルメディア再生）のサポート、およびキャストデバイス 1 0 6 上のLEDの制御のサポートを含んでもよい。

40

## 【0 1 2 0】

図 5 は、いくつかの実装形態に係る、デバイス上の口頭入力を処理するための方法 5 0 0 のフロー図である。方法 5 0 0 は、オーディオ入力システム（たとえば、オーディオ入力装置 1 0 8 / 1 3 2）と、1つ以上のプロセッサ（たとえば、処理装置（複数可）2 0 2）と、1つ以上のプロセッサによって実行される1つ以上のプログラムを格納するメモリ（たとえば、メモリ 2 0 6）とを有する電子デバイス（たとえば、音声アシスタントク

50

クライアントデバイス 104、キャストイングデバイス 106)において実行される。いくつかの実装形態では、電子デバイスは、オーディオ入力システム(たとえば、オーディオ入力装置 108 / 132)と、1つ以上のプロセッサ(たとえば、処理装置(複数可) 202)と、1つ以上のプロセッサによって実行される1つ以上のプログラムを格納したメモリ(たとえば、メモリ 206)とを備え、当該1つ以上のプログラムは、方法 500を実行するための命令を含む。いくつかの実装形態では、非一時的なコンピュータ読み取り可能な記憶媒体が1つ以上のプログラムを含み、当該1つ以上のプログラムは、命令を含み、当該命令は、オーディオ入力システム(たとえば、オーディオ入力装置 108 / 132)と、1つ以上のプロセッサ(たとえば、処理装置(複数可) 202)とを有する電子デバイスによって実行されると、電子デバイスに、方法 500を実行させる。方法 500を実行するためのプログラムまたは命令は、図 2 ~ 図 4を参照して上述されたモジュール、ライブラリなどに含まれてもよい。

10

**【0121】**

デバイスは、デバイスにおいて口頭入力を受信する(502)。クライアントデバイス 104 / キャスティングデバイス 106は、ユーザが発した口頭入力(たとえば、音声入力)をキャプチャする。

**【0122】**

デバイスは、口頭入力を処理する(504)。クライアントデバイス 104 / キャスティングデバイス 106が口頭入力を処理する。処理には、ホットワード検出、テキストデータへの変換、およびユーザが提供するコマンド、要求、および/またはパラメータに対応する単語および表現の識別が含まれてもよい。いくつかの実装形態では、この処理は、最低限であってもよく、または、処理がまったくなくてもよい。たとえば、この処理は、サーバ 114に送信するために口頭入力オーディオをエンコードすることを含んでもよく、または、サーバ 114に送信するために口頭入力のキャプチャされた生オーディオを用意することを含んでもよい。

20

**【0123】**

デバイスは、口頭入力に基づいて決定された情報を含む要求を、遠隔システムに送信する(506)。クライアントデバイス 104 / キャスティングデバイス 106は、口頭入力を処理し、口頭入力から要求および1つ以上の関連するパラメータを識別することによって、口頭入力から要求を決定する。クライアントデバイス 104 / キャスティングデバイス 106は、決定した要求を遠隔システム(たとえば、サーバ 114)に送信する。遠隔システムは、要求に対する応答を決定および生成する。いくつかの実装形態では、クライアントデバイス 104 / キャスティングデバイス 106は、口頭入力を(たとえば、エンコードされたオーディオとしての、生オーディオデータとしての)サーバ 114に送信し、サーバ 114は、口頭入力を処理し、要求および関連するパラメータを決定する。

30

**【0124】**

デバイスは、要求に対する応答を受信する(508)。応答は、口頭入力に基づいた情報に応じて、遠隔システムによって生成されてもよい。遠隔システム(たとえば、サーバ 114)は、要求に対する応答を決定および生成し、この応答をクライアントデバイス 104 / キャスティングデバイス 106に送信する。

40

**【0125】**

デバイスは、応答に応じて動作を実行する(510)。クライアントデバイス 104 / キャスティングデバイス 106が、受信した応答に応じて、1つ以上の動作を実行する。たとえば、応答が、デバイスに特定の情報をオーディオによって出力させるためのコマンドである場合、クライアントデバイス 104 / キャスティングデバイス 106は、この情報を取り出し、この情報を音声オーディオ出力に変換し、音声オーディオをスピーカから出力する。別の例として、応答が、デバイスにメディアコンテンツを再生させるためのコマンドである場合、クライアントデバイス 104 / キャスティングデバイス 106は、メディアコンテンツを取り出し、メディアコンテンツを再生する。

**【0126】**

50



前記した、受信すること、処理すること、送信すること、受信すること、および実行することのうちの1つ以上は、電子デバイス上で実行中の音声アシスタントライブラリの1つ以上の音声処理モジュールによって行われ、音声処理モジュールは、電子デバイス上で実行中または実行可能な1つ以上のアプリケーションプログラムおよび/またはオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供する(512)。クライアントデバイス104/キャストデバイス106は、前記した、受信するステップ、処理するステップ、送信するステップ、受信するステップ、および実行するステップのうちの1つ以上を実行するための関数およびモジュールを含む音声アシスタントクライアントライブラリ240を有してもよい。音声アシスタントクライアントライブラリ240のモジュールは、ライブラリ240を含むまたはライブラリ240にリンクする(たとえば、ライブラリ240および関連APIを実行する)クライアントデバイス104/キャストデバイス106におけるアプリケーション、オペレーティングシステム、およびプラットフォームソフトウェアにアクセス可能な複数の音声処理動作およびアシスタント動作を提供する。

#### 【0127】

いくつかの実装形態では、音声処理モジュールに関連付けられた少なくともいくつかの音声処理動作は、ワイドエリアネットワークを介して電子デバイスと互いに接続される遠隔システム上で行われてもよい。たとえば、要求を決定するために口頭入力を処理することは、ネットワーク(複数可)112を通してクライアントデバイス104/キャストデバイス106と接続されたサーバ114によって行われてもよい。

#### 【0128】

いくつかの実装形態では、音声アシスタントライブラリは、複数の異なるデバイスタイプ上で動作可能な共通のオペレーティングシステム上で実行可能であり、それによって、音声処理動作のうちの1つ以上と対話するように構成された音声対応アプリケーションのポータビリティを可能にする。音声アシスタントクライアントライブラリ240(ならびに、関連ライブラリおよびAPI、たとえば、デバッグライブラリ242、プラットフォームAPI244、POSIX API246)は、定義済みのオペレーティングシステム(たとえば、Linux)の標準要素(たとえば、オブジェクト)を利用するので、定義済みのオペレーティングシステムのディストリビューションまたはフレーバー(たとえば、異なるLinuxまたはLinuxベースのディストリビューションまたはフレーバー)を実行するいろいろなデバイス上で動作可能である。このように、音声アシスタント機能をいろいろなデバイスが利用可能であり、音声アシスタントエクスペリエンスは、当該いろいろなデバイス間で一貫している。

#### 【0129】

いくつかの実装形態では、要求および応答は、デバイスにおいて処理されてもよい。たとえば、タイマ、目覚まし時計、時計、および音量調節など、デバイスにローカルであり得る基本関数については、クライアントデバイス104/キャストデバイス106が、口頭入力を処理し、これらの基本関数のうちの1つに要求が対応すると判断し、デバイスにおいて応答を決定し、応答に応じて1つ以上の動作を実行してもよい。デバイスは、ログを取る目的のためにサーバ114に引き続き要求および応答を報告してもよい。

#### 【0130】

いくつかの実装形態では、オーディオ入力システムを備える電子デバイス用の、デバイスを問わない音声アシスタントライブラリは、複数の異なる電子デバイスタイプ上に実装される共通のオペレーションシステム上で実行するように構成された1つ以上の音声処理モジュールを含み、当該音声処理モジュールは、電子デバイス上で実行中のアプリケーションプログラムおよびオペレーティングソフトウェアにアクセス可能な複数の音声処理動作を提供し、それによって、音声処理動作のうちの1つ以上と対話するように構成された音声対応アプリケーションのポータビリティが可能になる。音声アシスタントクライアントライブラリ240は、同じ定義済みのオペレーティングシステムベースをライブラリ(たとえば、ライブラリおよびデバイスのオペレーティングシステムは、Linuxベース

10

20

30

40

50

である)として共有するいろいろなデバイス上で実行され得るライブラリであるため、このライブラリは、デバイスを問わない。ライブラリ240は、いろいろなデバイス間でアプリケーションにアクセス可能な音声アシスタント機能のための複数のモジュールを提供する。

#### 【0131】

いくつかの実装形態では、音声処理モジュールに関連付けられた少なくともいくつかの音声処理動作は、ワイドエリアネットワークを介して電子デバイスと互いに接続されるバックエンドサーバ上で実行される。たとえば、ライブラリ240は、サーバ114と通信し、口頭入力を処理するためにサーバ114に送信し、要求を決定するモジュールを含む。

10

#### 【0132】

いくつかの実装形態では、音声処理動作は、電子デバイスに(たとえば、直接または通信可能に)接続されたデバイスを制御するように構成されたデバイス固有の動作を含む。ライブラリ240は、クライアントデバイス104/キャストデバイス106に接続されたその他のデバイス(たとえば、ワイヤレススピーカ、スマートテレビなど)を制御するための関数またはモジュールを含んでもよい。

#### 【0133】

いくつかの実装形態では、音声処理動作は、要求された情報および/またはメディアコンテンツを電子デバイスのユーザに提供する、または電子デバイスと(たとえば、直接または通信可能に)接続されたデバイス上で提供するように構成された情報/メディア要求動作を含む。ライブラリ240は、情報またはメディアを取り出して、情報またはメディアをクライアントデバイス104/キャストデバイス106上または接続されたデバイス上で提供する(たとえば、電子メールを読み上げる、新聞記事を読み上げる、ストーリーミング音楽を再生する)ための関数またはモジュールを含んでもよい。

20

#### 【0134】

様々な要素を説明するために、用語「第1の」、「第2の」などが本明細書において使用され得るが、要素は、これらの用語によって限定されるべきではないと理解されるだろう。これらの用語は、1つの要素を別の要素と区別するために使用されるにすぎない。たとえば、第1のコンタクトの名称がすべて矛盾なく変更され、第2のコンタクトの名称がすべて矛盾なく変更される場合に限り、説明の意味を変更することなく、第1のコンタクトを第2のコンタクトと称することができ、同様に、第2のコンタクトを第1のコンタクトと称することができる。第1のコンタクトおよび第2のコンタクトは、両方ともコンタクトであるが、同じコンタクトではない。

30

#### 【0135】

本明細書において使用される用語は、特定の実装形態を説明するためだけのものであり、特許請求の範囲を限定することを意図していない。実装形態および添付の請求の範囲の説明において使用される単数形「a」、「an」、および「the」は、文脈が明らかに他を示さない限り、複数形も含むことを意図する。本明細書において使用される用語「および/または(and/or)」は、関連する記載された項目のうちの1つ以上のいずれか、およびすべての考えられる組み合わせを指し、それらを包含すると理解されるだろう。用語「備える/含む(comprises)」および/または「備える/含む(comprising)」は、本明細書において使用されるとき、記載の特徴、整数、ステップ、動作、要素、および/または構成要素の存在を具体的に挙げるが、1つ以上のその他の特徴、整数、ステップ、動作、要素、構成要素、および/またはそれらの群の存在もしくは追加を排除しないと理解されるだろう。

40

#### 【0136】

本明細書で使用するとき、用語「~である場合(if)」は、文脈に応じて、記載の先行条件が真である「ときに(when)」、「すると(upon)」、「と判定することに応答して(in response to determining)」、「という判定に応じて(in accordance with a determination)」または「と検出することに応答して(in response to detecting)」を意

50

味すると解釈され得る。同様に、表現「記載の先行条件が真であると判定された場合（if it is determined[that a stated condition precedent is true]）」、「記載の先行条件が真である場合（if[a stated condition precedent is true]）」、「記載の先行条件が真であるとき（when[a stated condition precedent is true]）」は、文脈に応じて、記載の先行条件が真である「と判定すると（upon determining）」、「と判定すること」に  
 応答して（in response to determining）」、「という判定に応じて（in accordance with a determination）」、「と検出すると（upon detecting）」、または「と検出すること」に  
 応答して（in response to detecting）」を意味すると解釈され得る。

#### 【 0 1 3 7 】

様々な実装形態を詳細に参照し、その例を添付の図面に示す。以下の詳細な説明において、本発明および記載の実装形態の十分な理解のため、たくさんの具体的な詳細を記載した。しかしながら、これらの具体的な詳細がなくても、本発明を実施することができる。その他の場合、周知の方法、プロシージャ、構成要素、および回路については、実装形態の態様を不必要にあいまいにしないよう、詳細に説明しなかった。

#### 【 0 1 3 8 】

上記の説明は、説明の便宜上、具体的な実装形態を例に記載された。しかしながら、上記例示の説明は、網羅的であったり、開示の厳密な形態に本発明を限定したりすることを意図しない。上記教示に鑑みて、多くの変更例および変形例が可能である。当業者が、考えられる特定の用途に適した様々な変更例を用いて本発明および様々な実装形態を最大限に利用することを可能にするために、実装形態は、本発明の原理およびその実際の適用を最もよく説明するために選択および記載されている。

【 図 1 】

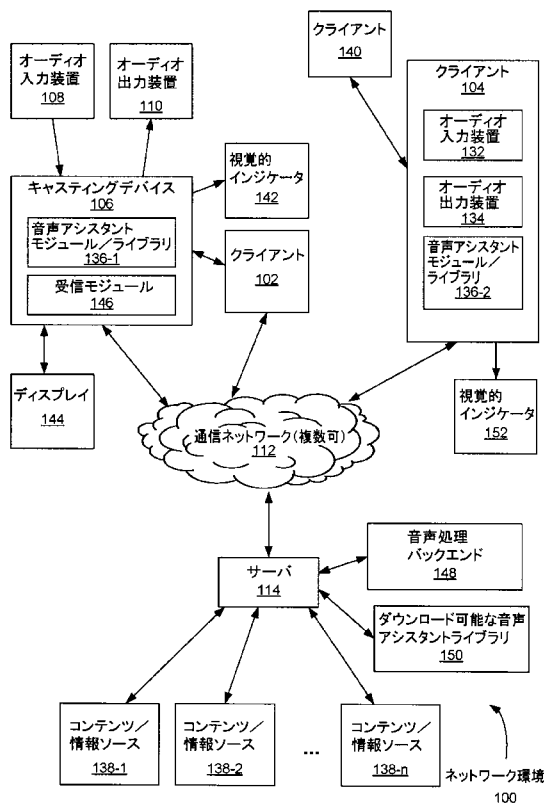


FIGURE 1

【 図 2 】

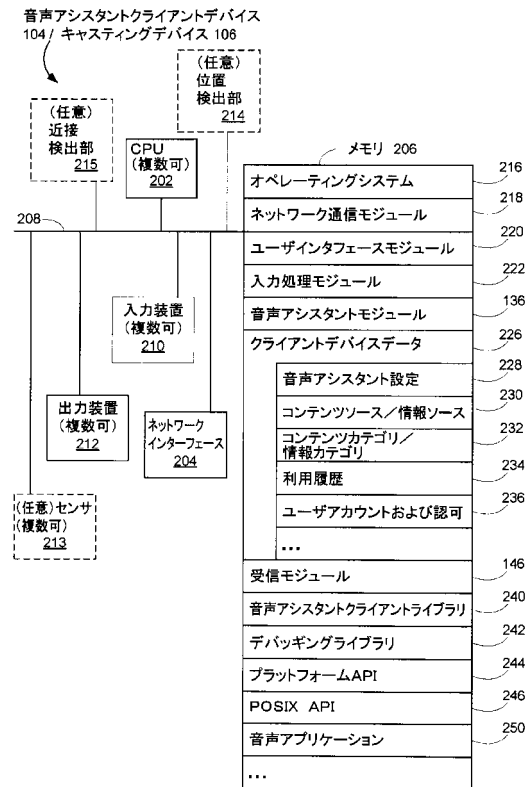
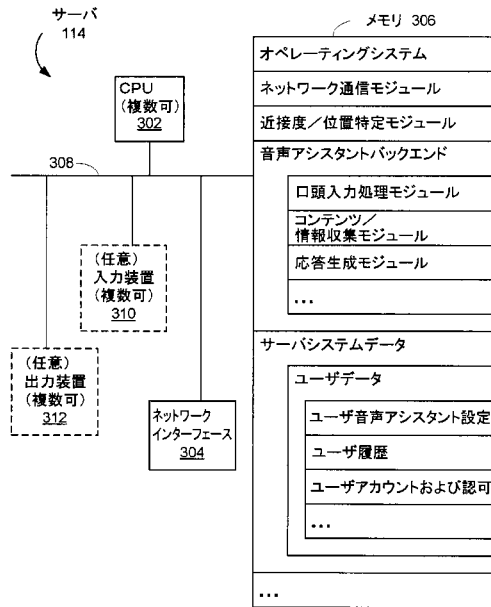


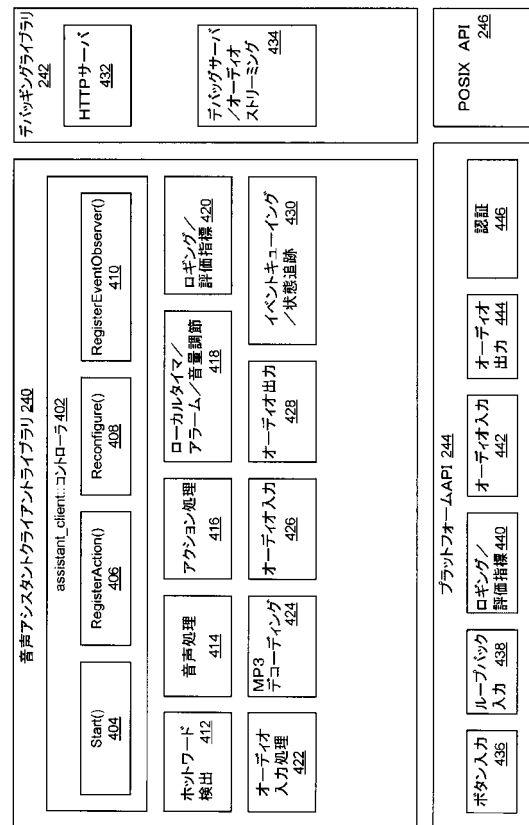
FIGURE 2

【 図 3 】



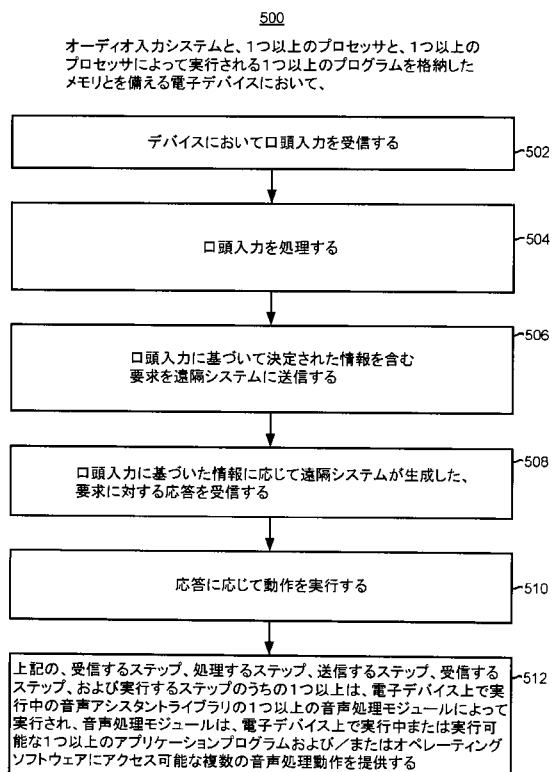
**FIGURE 3**

【 図 4 】



## FIGURE 4

【 図 5 】



**Figure 5**

## 【 国際調査報告 】

## INTERNATIONAL SEARCH REPORT

International application No

PCT/US2017/032002

## A. CLASSIFICATION OF SUBJECT MATTER

INV. G06F3/16  
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	WO 2012/103321 A2 (AMAZON TECH INC [US]; WILLIAMS HUNTER [US]; CARR WILLIAM D [US]; JAMES) 2 August 2012 (2012-08-02) paragraph [0008] - paragraph [0053]; figures 1-4	1-15
A	----- US 2014/365226 A1 (SINHA ANOOP K [US]) 11 December 2014 (2014-12-11) paragraphs [0032] - [0040]; figure 1	1-15
A	----- US 2014/006483 A1 (GARMAK STEN [SE] ET AL) 2 January 2014 (2014-01-02) paragraphs [0029] - [0073]; figure 1.2	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

\*&amp;\* document member of the same patent family

Date of the actual completion of the international search

18 August 2017

Date of mailing of the international search report

25/08/2017

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

Arranz, José

**INTERNATIONAL SEARCH REPORT**

Information on patent family members

International application No

PCT/US2017/032002

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2012103321 A2	02-08-2012	CN 103620546 A EP 2668561 A2 JP 2014507030 A US 2012198339 A1 WO 2012103321 A2	05-03-2014 04-12-2013 20-03-2014 02-08-2012 02-08-2012
US 2014365226 A1	11-12-2014	US 2014365226 A1 WO 2014197336 A1	11-12-2014 11-12-2014
US 2014006483 A1	02-01-2014	EP 2868060 A2 EP 2868061 A2 EP 2999191 A1 US 2014006483 A1 US 2015199122 A1 US 2016191574 A1 US 2017230429 A1 WO 2014001912 A2 WO 2014001913 A2	06-05-2015 06-05-2015 23-03-2016 02-01-2014 16-07-2015 30-06-2016 10-08-2017 03-01-2014 03-01-2014

## フロントページの続き

(51) Int.Cl. F I テーマコード ( 参考 )  
**G 0 6 Q 50/10 (2012.01) G 0 6 Q 50/10**

- (31)優先権主張番号 62/336,551  
 (32)優先日 平成28年5月13日(2016.5.13)  
 (33)優先権主張国・地域又は機関  
 米国(US)
- (31)優先権主張番号 62/336,565  
 (32)優先日 平成28年5月13日(2016.5.13)  
 (33)優先権主張国・地域又は機関  
 米国(US)
- (31)優先権主張番号 62/336,569  
 (32)優先日 平成28年5月13日(2016.5.13)  
 (33)優先権主張国・地域又は機関  
 米国(US)

(81)指定国・地域 AP(BW, 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, MC, 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, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, 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, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ