(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 108604179 A
(43)申请公布日 2018.09.28

(21)申请号 201780009238.2

(22)申请日 2017.05.10

(30)优先权数据
62/334,434 2016.05.10 US
62/336,566 2016.05.13 US
62/336,551 2016.05.13 US
62/336,565 2016.05.13 US
62/336,569 2016.05.13 US

(85)PCT国际申请进入国家阶段日
2018.08.01

(86)PCT国际申请的申请数据
PCT/US2017/032002 2017.05.10

(87)PCT国际申请的公布数据
WO2017/197010 EN 2017.11.16

(71)申请人 谷歌有限责任公司
地址 美国加利福尼亚州

(72)发明人 肯尼斯·米克斯特

(74)专利代理机构 中原信达知识产权代理有限责任公司 11219
代理人 周亚荣 安翔

(51)Int.Cl.
G06F 3/16(2006.01)

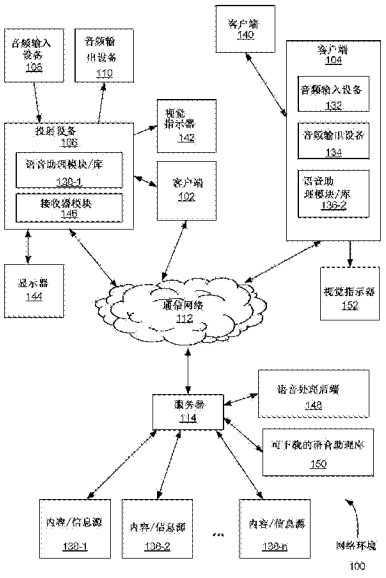
权利要求书2页 说明书31页 附图5页

(54)发明名称

设备上语音助理的实现

(57)摘要

在具有音频输入系统的电子设备处的方法，包括：接收设备处的口头输入；处理口头输入；向远程系统传送请求，所述请求包括基于口头输入确定的信息；接收对请求的响应，其中所述响应是由远程系统根据基于口头输入的信息生成的；以及根据响应执行操作，其中所述接收、处理、传送、接收和执行中的一个或多个由在电子设备上执行的语音助理库的一个或多个语音处理模块来执行，所述语音处理模块提供对在电子设备上执行或可执行的一个或多个应用程序和/或操作软件可访问的多个语音处理操作。



CN 108604179 A

1.一种方法,包括:

在包括音频输入系统、一个或多个处理器和存储用于由所述一个或多个处理器执行的一个或多个程序的存储器的电子设备处:

接收所述设备处的口头输入;

处理所述口头输入;

向远程系统传送请求,所述请求包括基于所述口头输入确定的信息;

接收对所述请求的响应,其中所述响应是由所述远程系统根据基于所述口头输入的所述信息生成的;以及

根据所述响应执行操作,

其中所述接收、处理、传送、接收和执行中的一个或多个由在所述电子设备上执行的语音助理库的一个或多个语音处理模块来执行,所述语音处理模块提供对在所述电子设备上执行或能够执行的一个或多个应用程序和/或操作软件能够访问的多个语音处理操作。

2.根据权利要求1所述的方法,其中与所述语音处理模块相关联的至少一些语音处理操作在所述远程系统上执行,所述远程系统经由广域网与所述电子设备互连。

3.根据任一前述权利要求所述的方法,其中所述语音助理库是在多个不同设备类型上能够操作的公共操作系统上能够执行的,由此实现被配置成与所述语音处理操作中的一个或多个交互的语音使能应用的可移植性。

4.一种用于包括音频输入系统的电子设备的设备不可知的语音助理库,包括:

一个或多个语音处理模块,所述一个或多个语音处理模块被配置成在多个不同的电子设备类型上实现的公共操作系统上执行,所述语音处理模块提供多个语音处理操作,所述多个语音处理操作是对在所述电子设备上执行的应用程序和操作软件能够访问的,由此实现被配置成与所述语音处理操作中的一个或多个交互的语音使能应用的可移植性。

5.根据任一前述权利要求所述的语音助理库,其中与所述语音处理模块相关联的至少一些语音处理操作在经由广域网与所述电子设备互连的后端服务器上执行。

6.根据任一前述权利要求所述的语音助理库,其中所述语音处理操作包括被配置成控制与所述电子设备耦合的设备的特定于设备的操作。

7.根据任一前述权利要求所述的语音助理库,其中所述语音处理操作包括信息和媒体请求操作,所述信息和媒体请求操作被配置成向所述电子设备的用户或与所述电子设备耦合的设备上的用户提供所请求的信息和/媒体内容。

8.一种电子设备,包括:

音频输入系统;

一个或多个处理器;和

存储器,所述存储器存储要由所述一个或多个处理器执行的一个或多个程序,所述一个或多个程序包括用于以下的指令:

接收在所述设备处的口头输入;

处理所述口头输入;

向远程系统传送请求,所述请求包括基于所述口头输入确定的信息;

接收对所述请求的响应,其中所述响应是由所述远程系统根据基于所述口头输入的所述信息生成的;以及

根据所述响应执行操作，

其中所述接收、处理、传送、接收和执行中的一个或多个由在所述电子设备上执行的语音助理库的一个或多个语音处理模块来执行，所述语音处理模块提供对在所述电子设备上执行或能够执行的一个或多个应用程序和/或操作软件能够访问的多个语音处理操作。

9.根据权利要求8所述的设备，其中与所述语音处理模块相关联的至少一些语音处理操作在所述远程系统上执行，所述远程系统经由广域网与所述电子设备互连。

10.根据任一前述权利要求所述的设备，其中所述语音助理库是在多个不同设备类型上能够操作的公共操作系统上能够执行的，由此实现被配置成与所述语音处理操作中的一个或多个交互的语音使能应用的可移植性。

11.一种存储一个或多个程序的非暂时性计算机可读存储介质，所述一个或多个程序包括指令，所述指令在由具有音频输入系统和一个或多个处理器的电子设备执行时使所述电子设备：

接收在所述设备处的口头输入；

处理所述口头输入；

向远程系统传送请求，所述请求包括基于所述口头输入确定的信息；

接收对所述请求的响应，其中所述响应是由所述远程系统根据基于所述口头输入的所述信息生成的；以及

根据所述响应执行操作，

其中所述接收、处理、传送、接收和执行中的一个或多个由在所述电子设备上执行的语音助理库的一个或多个语音处理模块来执行，所述语音处理模块提供对在所述电子设备上执行或能够执行的一个或多个应用程序和/或操作软件能够访问的多个语音处理操作。

12.根据权利要求11所述的计算机可读存储介质，其中与所述语音处理模块相关联的至少一些语音处理操作在所述远程系统上执行，所述远程系统经由广域网与所述电子设备互连。

13.根据任一前述权利要求所述的计算机可读存储介质，其中所述语音助理库是在多个不同设备类型上能够操作的公共操作系统上能够执行的，由此实现被配置成与所述语音处理操作中的一个或多个交互的语音使能应用的可移植性。

14.一种电子设备，包括：

音频输入系统；

一个或多个处理器；以及

存储器，所述存储器存储要由所述一个或多个处理器执行的一个或多个程序，所述一个或多个程序包括用于执行根据权利要求1-3中任一项所述的方法的指令。

15.一种存储一个或多个程序的非暂时性计算机可读存储介质，所述一个或多个程序包括指令，所述指令在由具有音频输入系统和一个或多个处理器的电子设备执行时使所述电子设备执行权利要求1-3中任一项所述的方法。

设备上语音助理的实现

技术领域

[0001] 本申请大体涉及计算机技术，包括但不限于用于设备的语音助理和相关库。

背景技术

[0002] 通过音频/语音输入和输出与用户交互的基于语音的助理随着互联网和云计算的发展而日益普及。这些助理可以为数字媒体消费提供界面，并提供各种类型的信息，包括新闻、体育比分、天气和股票等等。

[0003] 用户可以具有期望基于语音的助理功能的多个设备。期望有基于语音的助理，其可以跨各种设备实施和使用、可以跨各种设备提供一致的体验、并且可以支持特定于特定设备的功能。

发明内容

[0004] 本说明书中描述的实施方式旨在以实现控制用于各种操作系统平台的本地设备的方式在嵌入式系统和/或设备中嵌入或包括语音助理。

[0005] 根据一些实施方式，精简的、低资源使用设备侧库具有包括本地处理音频数据、侦听唤醒词(wakewords)或热词(hotwords)以及发送用户请求的特征。其他功能还包括与云大脑、可扩展的语音动作控制系统、可允许集成到多种不同的操作环境中的移植性层的连接，以及与其余的客户端软件异步更新的能力。

[0006] 所描述的实施方式具有提供用于在许多不同设备上与语音助理进行交互的类似用户体验的优点。

[0007] 所描述的实施方式具有另一优点，实现在来自从设备本身使能的创新中的语音助理能力中的解耦创新。例如，如果创建了改进的识别管道，它可以推送到设备，而设备制造商不需要做任何事情来接收它，并且仍然可以从先前的语音命令中受益。

[0008] 根据一些实施方式，在具有音频输入系统、一个或多个处理器以及存储由一个或多个处理器执行的一个或多个程序的存储器的电子设备处的方法包括：接收设备处的口头输入；处理口头输入；向远程系统传送请求，所述请求包括基于口头输入确定的信息；接收对请求的响应，其中所述响应是由远程系统根据基于口头输入的信息生成的；以及根据所述响应执行操作，其中所述接收、处理、传送、接收和执行中的一个或多个由在电子设备上执行的语音助理库的一个或多个语音处理模块来执行，所述语音处理模块提供对在电子设备上执行或可执行的一个或多个应用程序和/或操作软件可访问的多个语音处理操作。

[0009] 在一些实施方式中，用于包括音频输入系统的电子设备的设备不可知语音助理库包括：一个或多个语音处理模块，其被配置为在多个不同电子设备类型上实现的公共操作系统上执行，所述语音处理模块提供在电子设备上执行的应用程序和操作软件可访问的多个语音处理操作，由此实现被配置为与一个或多个语音处理操作交互的语音使能应用的可移植性。

[0010] 在一些实施方式中，电子设备包括音频输入系统，一个或多个处理器以及存储要

由一个或多个处理器执行的一个或多个程序的存储器。一个或多个程序包括用于执行以下步骤的指令:接收设备处的口头输入;处理口头输入;向远程系统传送请求,所述请求包括基于口头输入确定的信息;接收对请求的响应,其中所述响应是由远程系统根据基于口头输入的信息生成的;以及根据响应执行操作,其中所述接收、处理、传送、接收和执行中的一个或多个由在电子设备上执行的语音助理库的一个或多个语音处理模块来执行,所述语音处理模块提供对在电子设备上执行或可执行的一个或多个应用程序和/或操作软件可访问的多个语音处理操作。

[0011] 在一些实施方式中,非暂时性计算机可读存储介质存储一个或多个程序。一个或多个程序包括当由具有音频输入系统和一个或多个处理器的电子设备执行时使所述电子设备执行以下操作的指令接收设备处的口头输入;处理口头输入;向远程系统传送请求,所述请求包括基于口头输入确定的信息;接收对请求的响应,其中所述响应是由远程系统根据基于口头输入的信息生成的;以及根据响应执行操作,其中所述接收、处理、传送、接收和执行中的一个或多个由在电子设备上执行的语音助理库的一个或多个语音处理模块来执行,所述语音处理模块提供对在电子设备上执行或可执行的一个或多个应用程序和/或操作软件可访问的多个语音处理操作。

附图说明

[0012] 图1是示出根据一些实施方式的示例网络环境的框图。

[0013] 图2是示出根据一些实施方式的示例语音助理客户端设备的图。

[0014] 图3是示出根据一些实施方式的示例服务器系统的图。

[0015] 图4是示出根据一些实施方式的语音助理库的功能视图的框图。

[0016] 图5是根据一些实施方式的用于处理设备上的口头输入的方法的流程图。

[0017] 在整个附图中,相同的附图标记指代对应的部件。

具体实施方式

[0018] 现在将详细参照各种实施方式,其示例在附图中示出。在以下具体实施方式中,阐述了许多具体细节以提供对本发明和所描述的实施方式的透彻理解。然而,可以在没有这些具体细节的情况下实践本发明。在其他实例中,未详细描述众所周知的方法、过程、组件和电路,以免不必要地模糊实施方式的方面。

[0019] 在一些实施方式中,语音助理的目的是向用户提供在各种设备上都可用的个性化语音界面,并且启用各种各样的用例,在整个用户日提供一致的体验。语音助理和/或相关功能可以被集成到第一方和第三方产品和设备中。

[0020] 示例用例涉及媒体。语音命令可用于通过语音启动对音乐、收音机、播客、新闻和其他音频媒体的回放和控制。例如,用户可以发出语音命令(例如“播放爵士乐(play jazz music)”、“播放107.5FM(play 107.5FM)”、“跳到下一首歌曲(skip to next song)”、“播放‘系列音乐’(play‘Serial’)”)来播放或控制各种类型的音频媒体。此外,这样的命令可以用于播放来自各种来源的音频媒体,诸如地面无线电台的在线流媒体、音乐订阅服务、本地存储、远程存储等。此外,语音助理可以利用投射设备(casting device)可用的集成来支持附加内容。

[0021] 另一个示例用例涉及远程回放。用户可以向包括语音助理功能的投射设备发出语音命令，并且根据语音命令，媒体被重放(例如，被投射到)在命令中指定的设备上、在一个或多个设备的指定组中的设备上、或者在命令中指定的区域中的一个或多个设备上。用户还可以在命令中指定通用类别或特定内容，并且根据命令中指定的类别或内容来播放适当的媒体。

[0022] 另一示例用例是非媒体，诸如生产率特征(例如定时器、警报闹钟、日历)、家庭自动化、由搜索引擎供电的问题和答案(例如，搜索查询)、有趣的(例如，助理人物、笑话、游戏、复活节彩蛋)和日常任务(如交通、导航、食品、金融、礼品等)。

[0023] 在一些实施方式中，语音助理被提供为投射设备的可选特征，并且语音助理功能可以被更新为投射设备的一部分。

[0024] 在一些实施方式中，应用处理器(例如，在用户说出语音命令或口头输入的客户端设备或投射设备处执行)执行来自用户的语音命令以及口头输入中的热词或关键词的检测。在一些实施方式中，通过外部数字信号处理器来执行(例如，与用户对其说出语音命令或口头输入的客户端或投射设备相反，由处理语音命令的服务器系统执行)热词的检测。

[0025] 在一些实施方式中，具有语音助理特征的设备包括以下中的一个或多个：远场支持、“即按即助(push to assist)”或“即按即说(push to talk)”(例如，发起语音助理功能的按钮)以及AC功率。

[0026] 在一些实施方式中，语音助理包括用于以下各项中的一个或多个的应用程序编程接口(API)：音频输入(例如麦克风，用于正在进行回放的媒体回送)、麦克风状态(例如开/关)、回避(例如，当通过热词或即按即说触发助理时，降低所有输出的音量)、以及新的助理事件和状态消息(例如，助理被触发(例如听到热词、按下助理按钮)、听话音、在服务器上等待、响应、响应完成、闹钟/定时器正在播放)。

[0027] 在一些实施方式中，具有语音助理功能的设备可以出于配置目的而与另一设备通信(例如，利用智能手机上的配置应用)，以启用或促进设备上的语音助理的功能(例如，在设备上设置语音助理功能，向用户提供教程)。配置或设置可以包括指定设备位置、与用户账户关联、用户选择进入语音控制、链接到媒体服务(例如，视频流传输服务、音乐流传输服务)并优先化、以及家庭自动化配置等。

[0028] 在一些实施方式中，具有语音助理的设备可以包括一个或多个用户界面元件或向用户的指示。一个或多个用户界面元件是物理的(例如，作为使用一个或多个LED显示的光图案、作为由扬声器输出的声音图案)，并且可以包括以下一个或多个：不依赖于热词的“即按即助”或“即按即助说”触发器，“静音麦克风(mute microphone)”触发器和视觉状态指示，“等待热词状态(awaiting hotword status)”视觉指示，“检测到的热词(hotword detected)”视觉指示，在一定距离(例如15英尺)处可见的“助理正在主动收听(assistant is actively listening)”视觉指示，“助理正在工作/思考(assistant is working/thinking)”视觉指示，“语音消息/通知可用(voice message/notification is available)”视觉指示，“音量级别(volume level)”控制方法和状态指示器，以及“暂停/恢复(pause/resume)”控制方法。在一些实施方式中，这些物理用户界面元件由客户端设备或投射设备提供。在一些实施方式中，语音助理支持在不同设备上通用的用户界面元件或指示集合，以便在不同设备上的体验的一致性。

[0029] 在一些实施方式中，语音助理支持特定于设备的命令和/或热词以及标准化的、预定义的命令和/或热词集合。

[0030] 图1示出了根据一些实施方式的网络环境100。网络环境100包括投射设备106和/或语音助理客户端设备104。投射设备106(例如，GOOGLE INC.的CHROMECAST)直接或以其他方式通信地耦合到音频输入设备108(例如，麦克风)和音频输出设备110(例如，一个或多个扬声器)。在一些实施方式中，音频输入设备108和音频输出设备110都是通信地耦合到投射设备106的设备(例如，扬声器系统、电视机、声音棒)的组件。在一些实施方式中，音频输入设备108是投射设备106的组件，并且音频输出设备110是投射设备106通信地耦合的设备的组件，反之亦然。在一些实施方式中，音频输入设备108和音频输出设备110是投射设备106的组件。

[0031] 在一些实施方式中，投射设备106通信地耦合到客户端102。客户端102可包括促进配置投射设备106的应用或模块(例如，投射设备设置应用程序)，包括语音助理特征。

[0032] 在一些实施方式中，投射设备106耦合到显示器144。

[0033] 在一些实施方式中，投射设备106包括一个或多个视觉指示器142(例如，LED灯)。

[0034] 在一些实施方式中，投射设备106包括接收器模块146。在一些实施方式中，例如，接收器模块146操作包括硬件功能和与内容源通信的投射设备106。在一些实施方式中，在投射设备106处存在用于不同内容源的不同接收器模块146。在一些实施方式中，接收器模块146包括用于不同内容源的相应子模块。

[0035] 语音助理客户端设备104(例如具有GOOGLE INC.的GOOGLE ASSISTANT，GOOGLE INC.的GOOGLE HOME的智能电话、膝上型计算机或台式计算机、平板计算机、语音命令设备、移动设备或车载系统)包括音频输入设备132(例如，麦克风)和音频输出设备134(例如，一个或多个扬声器、耳机)。在一些实施方式中，语音助理客户端设备104(例如，具有由GOOGLE INC.的GOOGLE ASSISTANT，GOOGLE INC.的GOOGLE HOME的语音命令设备、移动设备或车载系统)通信地耦合到客户端140(例如，智能手机、平板计算机设备)。客户端140可包括促进配置语音助理客户端设备104的应用或模块(例如，语音命令设备设置应用程序)，包括语音助理特征。

[0036] 在一些实施方式中，语音助理客户端设备104包括一个或多个视觉指示器152(例如，LED灯)。具有可视指示器(例如，LED灯)的语音助理客户端设备的示例在2016年5月13日提交的名称为“LED Design Language for Visual Affordance of Voice User Interfaces”的美国暂时申请No.62/336,566的图4A中示出，该申请通过引用整体并入本文。

[0037] 投射设备106和语音助理客户端设备104包括语音助理模块或库136的相应实例。语音助理模块/库136是跨各种设备(例如，投射设备106、语音助理客户端设备104)实现语音助理功能的模块/库。语音助理功能在各种设备中保持一致，同时仍实现特定于设备的特征(例如，通过语音助理支持控制特定于设备的特征)。在一些实施方式中，语音助理模块或库136在设备之间是相同的或相似的；相同库的实例可以包含在各种设备中。

[0038] 在一些实施方式中，取决于设备的类型，语音助理模块/库136被包括在安装到设备中、在设备操作系统中或嵌入设备中(例如嵌入在固件中)的应用中。

[0039] 在一些实施方式中，投射设备106处的语音助理模块/库136-1与接收器模块146通

信以执行语音助理操作。

[0040] 在一些实施方式中，投射设备106处的语音助理模块/库136-1可以控制或以其他方式影响视觉指示器142。

[0041] 在一些实施方式中，语音助理客户端设备104处的语音助理模块/库136-2可以控制或以其他方式影响视觉指示器152。

[0042] 投射设备106和语音助理客户端设备104通过一个或多个通信网络112(例如局域网、广域网、因特网)通信地耦合到服务器系统114。语音助理模块/库136检测(例如接收)由音频输入设备108/132拾取(例如，捕获)的口头输入，处理口头输入(例如，检测热词)，并且将处理后的口头输入或将处理后的口头输入编码传送到服务器114。服务器114接收处理后的口头输入或其编码，并处理所接收的口头输入以确定对口头输入的适当响应。适当响应可以是对投射设备106或语音助理客户端设备104执行功能或操作的内容、信息或指令或命令或元数据。服务器114将该响应发送到投射设备106或语音助理客户端设备104，在那里输出内容或信息(例如，通过音频输出设备110/134输出)和/或执行功能。作为处理的一部分，服务器114可以与一个或多个内容或信息源138通信以获得用于响应的内容或信息，或对其的引用。在一些实施方式中，内容或信息源138例如包括搜索引擎、数据库、与用户的账户相关联的信息(例如日历、任务列表、电子邮件)、网站和媒体流传输服务。在一些实施方式中，语音助理客户端设备104和投射设备106可以彼此通信或交互。2016年5月13日提交的名称为“LED Design Language for Visual Affordance of Voice User Interfaces”的美国临时申请No.62/336,566，2016年5月13日提交的名称为“Voice-Controlled Closed Caption Display”的美国临时申请No.62/336,569和2016年5月13日提交的名称为“Media Transfer among Media Output Devices”的美国临时申请No.62/336,565中描述了这种通信或交互的示例以及语音助理客户端设备104的示例操作(例如，GOOGLE INC.的GOOGLE HOME)，所有这些通过引用整体并入本文。

[0043] 在一些实施方式中，语音助理模块/库136接收由音频输入设备108/132捕获的口头输入并且将口头输入(没有或很少处理)或其编码传送到服务器114。服务器114处理口头输入以检测热词，确定适当的响应，并且将该响应发送给投射设备106或语音助理客户端设备104。

[0044] 如果服务器114确定口头输入包括用于投射设备106或语音助理客户端设备104执行功能的命令，则服务器114在响应中传送指令或元数据，该指令或元数据指示投射设备106或语音助理客户端设备104执行该功能。该功能可以是该设备特有的，并且用于在语音助理中支持这些功能的能力可以作为添加或链接到语音助理模块/库136的定制模块或功能而被包括在投射设备106或客户端104中。

[0045] 在一些实施方式中，服务器114包括或耦合到语音处理后端148，所述语音处理后端148执行口头输入处理操作并确定对口头输入的响应。

[0046] 在一些实施方式中，服务器114包括可下载的语音助理库150。可下载的语音助理库150(例如，与语音助理库136相同或其更新)可以包括新的特征和功能或更新，并且可以被下载以将语音助理库添加到设备或更新语音助理库136。

[0047] 图2是示出根据一些实施方式的网络环境100的示例语音助理客户端设备104或者投射设备106的框图。语音助理客户端设备104的示例包括但不限于移动电话、平板计算机、

膝上型计算机、台式计算机、无线扬声器(例如,GOOGLE INC.的GOOGLE HOME)、语音命令设备(例如,GOOGLE INC.的GOOGLE HOME)、电视、条形音箱、投射设备(例如GOOGLE INC.的CHROMECAST)、媒体流设备、家用电器、消费电子设备、车载系统、和可穿戴的个人设备。语音助理客户端设备104(例如,GOOGLE INC.的GOOGLE HOME,具有GOOGLE ASSISTANT能力的移动设备)或投射设备106(例如,GOOGLE INC.的CHROMECAST)通常包括一个或多个处理单元(CPU)202,网络接口204,存储器206以及用于互连这些组件的一个或多一个或多个通信总线208(有时称为芯片集)。语音助理客户端设备104或投射设备106包括促进用户输入的一个或多个输入设备210,其包括音频输入设备108或132(例如,语音命令输入单元或麦克风)以及可选的其他输入设备,诸如键盘、鼠标、触摸屏显示器、触敏输入板、手势捕获相机或其他输入按钮或控件)。在一些实施方式中,语音助理客户端设备102使用麦克风和语音识别或相机和手势识别来补充或替代键盘。语音助理客户端设备104或投射设备106还包括一个或多个输出设备212,其包括音频输出设备110或134(例如,一个或多个扬声器,耳机等),以及可选的一个或多个视觉显示器(例如,显示器144)和/或能够呈现用户界面并显示内容和信息的一个或多个视觉指示器142或152(例如,LED)。可选地,语音助理客户端设备104或者投射设备106包括用于确定语音助理客户端设备104或者投射设备106的位置的位置检测单元214,诸如GPS(全球定位卫星)或者其他地理位置接收器。语音助理客户端设备104或投射设备106可任选地还可以包括接近度检测设备215,例如IR传感器,用于确定语音助理客户端设备104或者投射设备106与其他对象(例如,用户,在穿戴式个人设备的情况下为穿着者)。可选地,语音助理客户端设备104或投射设备106包括传感器213(例如,加速度计、陀螺仪等)。

[0048] 存储器206包括高速随机存取存储器,诸如DRAM、SRAM、DDR RAM或其他随机存取固态存储器设备;并且可选地包括非易失性存储器,诸如一个或多个磁盘存储设备,一个或多个光盘存储设备,一个或多个闪存设备或一个或多个其他非易失性固态存储设备。存储器206可选地包括远离一个或多个处理单元202的一个或多个存储设备。存储器206或备选地存储器206内的非易失性存储器包括非暂时性计算机可读存储介质。在一些实施方式中,存储器206或存储器206的非暂时性计算机可读存储介质存储以下程序、模块和数据结构或其子集或超集:

[0049] • 操作系统216,包括用于处理各种基本系统服务和用于执行硬件相关任务的过程;

[0050] • 网络通信模块218,用于经由一个或多个网络接口204(有线或无线)以及一个或多个网络112(例如因特网、其他广域网、局域网、城域网等等)将语音助理客户端设备104或者投射设备106连接到其他设备(例如,服务器系统114,客户端102,140,其他语音助理客户端设备104或者投射设备106);

[0051] • 用户界面模块220,用于使得能够经由一个或多个输出设备212(例如,显示器、扬声器等)在语音助理客户端设备104或者投射设备106上呈现信息;

[0052] • 输入处理模块222,用于处理由一个或多个输入设备210捕获或接收的一个或多个用户输入或交互,并解释该输入或交互;

[0053] • 语音助理模块136,用于处理口头输入,向服务器114提供口头输入,接收来自服务器114的响应以及输出该响应;和

[0054] • 客户端数据226，用于存储至少与语音助理模块136相关联的数据，包括：

[0055] o语音助理设置228，用于存储与语音助理模块136和语音助理功能的设置和配置相关联的信息；

[0056] o内容/信息源230和类别232，用于存储预定义和/或用户指定的源以及内容或信息的类别；

[0057] o使用历史234，用于存储与语音助理模块136的操作和使用相关联的信息(例如，日志)，诸如接收到的命令和请求，对命令和请求的响应，响应于命令和请求而执行的操作等；和

[0058] o用户账户和授权236，用于存储一个或多个用户的授权和认证信息以访问内容/信息源230处的用户的相应账户以及这些授权账户的账户信息；和

[0059] o接收器模块146，用于操作投射设备106的投射功能，包括与内容源通信以接收用于回放的内容。

[0060] 在一些实施方式中，语音助理客户端设备104或者投射设备106包括一个或者多个库以及用于语音助理和相关功能的一个或者多个应用编程接口(API)。这些库可以被包括在语音助理模块136或接收器模块146中或由语音助理模块136或接收器模块146链接。该库包括与语音助理功能或促进语音助理功能的其他功能相关联的模块。API为硬件和其他软件(例如，操作系统、其他应用)提供接口，以促进语音助理功能。例如，语音助理客户端库240，调试库242，平台API 244和POSIX API 246可以存储在存储器206中。这些库和API在下面参照图4进一步描述。

[0061] 在一些实施方式中，语音助理客户端设备104或者投射设备106包括使用语音助理客户端库240的模块和功能的语音应用250，以及可选的调试库242，平台API 244和POSIX API 246。在一些实施方式中，语音应用250是通过使用语音助理客户端库240而语音使能的第一方或第三方应用等。

[0062] 每个上述识别的元件可以存储在一个或多个前述存储器设备中，并且对应于用于执行上述功能的指令集。以上识别的模块或程序(即，指令集)不需要被实现为单独的软件程序、过程、模块或数据结构，并且因此这些模块的各种子集可以在各种实施方式中被组合或以其他方式重新布置。在一些实施方式中，存储器206可选地存储以上识别的模块和数据结构的子集。此外，存储器206可选地存储上面没有描述的附加模块和数据结构。

[0063] 图3是示出根据一些实施方式的网络环境100的示例服务器系统114的框图。服务器114通常包括一个或多个处理单元(CPU)302，一个或多个网络接口304，存储器306以及用于互连这些组件(有时称为芯片集)的一个或多个通信总线308。服务器114可选地包括促进用户输入的一个或多个输入设备310，诸如键盘、鼠标、语音命令输入单元或麦克风、触摸屏显示器、触敏输入板、手势捕获照相机、或其他输入按钮或控件。此外，服务器114可以使用麦克风和语音识别或相机和手势识别来补充或替代键盘。在一些实施方式中，服务器114可选地包括一个或多个照相机，扫描仪或光传感器单元，用于捕获例如印制在电子设备上的图形系列代码的图像。服务器114可选地还包括使得能够呈现用户界面和显示内容的一个或多个输出设备312，包括一个或多个扬声器和/或一个或多个视觉显示器。

[0064] 存储器306包括高速随机存取存储器，诸如DRAM、SRAM、DDR RAM或其他随机存取固态存储器设备；并且可选地包括非易失性存储器，诸如一个或多个磁盘存储设备，一个或多

个光盘存储设备，一个或多个闪存设备或一个或多个其他非易失性固态存储设备。存储器306可选地包括远离一个或多个处理单元302定位的一个或多个存储设备。存储器306或备选地存储器306内的非易失性存储器包括非暂时性计算机可读存储介质。在一些实施方式中，存储器306或存储器306的非暂时性计算机可读存储介质存储以下程序、模块和数据结构或其子集或超集：

[0065] • 操作系统316，包括用于处理各种基本系统服务和用于执行硬件相关任务的过程；

[0066] • 网络通信模块318，用于经由一个或多个网络接口304(有线或无线)以及一个或多个网络112(例如因特网、其他广域网、局域网、城域网等等)将服务器系统114连接到其他设备(例如，语音助理客户端设备104，投射设备106，客户端102，客户端140)等；

[0067] • 接近度/位置确定模块320，用于基于客户端设备104或者投射设备106的位置信息来确定语音助理客户端设备104或者投射设备106的接近度和/或位置；

[0068] • 语音助理后端116，用于处理语音助理口头输入(例如，从语音助理客户端设备104和投射设备106接收的口头输入)，包括以下一项或多项：

[0069] o口头输入处理模块324，处理口头输入以识别口头输入中的命令和请求；

[0070] o内容/信息收集模块326，收集对命令和请求的内容和信息响应；和

[0071] o响应生成模块328，响应于命令和请求而产生口头输出并用响应内容和信息填充口头输出；和

[0072] • 服务器系统数据330，至少存储与语音助理平台的操作相关联的数据，包括：

[0073] o用户数据332，用于存储与语音助理平台的用户相关联的信息，包括：

[0074] • 用户语音助理设置334，用于存储对应于语音助理设置228的语音助理设置信息，以及对应于内容/信息源230和类别232的信息；

[0075] • 用户历史336，用于用语音助理(例如，日志)存储用户的历史，包括命令和请求的历史以及相应的响应；和

[0076] • 用户账户和授权338，用于存储用户的授权和认证信息以访问内容/信息源230处的用户的相应账户以及与用户账户和授权236相对应的那些授权账户的账户信息。

[0077] 每个上述识别的元件可以存储在一个或多个前述存储器设备中，并且对应于用于执行上述功能的指令集。以上识别的模块或程序(即，指令集)不需要被实现为单独的软件程序、过程、模块或数据结构，并且因此这些模块的各种子集可以在各种实施方式中被组合或以其他方式重新布置。在一些实施方式中，存储器306可选地存储以上识别的模块和数据结构的子集。此外，存储器306可选地存储上面没有描述的附加模块和数据结构。

[0078] 在一些实施方式中，语音助理模块136(图2)包括一个或多个库。库包括执行相应功能的模块或子模块。例如，语音助理客户端库包含执行语音助理功能的模块。语音助理模块136还可以包括用于与特定硬件(例如客户端或投射设备上的硬件)，特定操作软件或远程系统协作的一个或多个应用程序编程接口(API)。

[0079] 在一些实施方式中，库包括支持音频信号处理操作的模块，包括例如带通、滤波、擦除和热词检测。在一些实施方式中，库包括用于连接到后端(例如，基于服务器)的话音处理系统的模块。在一些实施方式中，库包括用于调试的模块(例如，调试话音识别、调试硬件问题、自动化测试)。

[0080] 图4示出了可以存储在语音助理客户端设备104或者投射设备106中并且由语音助理模块136或者另一应用运行的库和API。库和API可以包括语音助理客户端库240，调试库242，平台API 244和POSIX API246。在语音助理客户端设备104或者投射设备106(例如，语音助理模块136，可能希望支持与语音助理的协作的其他应用)可以包括或链接到库和API并运行以在应用中提供或语音使能助理功能。在一些实施方式中，语音助理客户端库240和调试库242是单独的库；保持语音助理客户端库240和调试库242的库分离促进解释这些库的不同安全含义的不同释放和更新过程。

[0081] 在一些实施方式中，库是柔性的；库可以在各种设备类型上使用，并包含相同的语音助理功能。

[0082] 在一些实施方式中，库依赖于标准共享对象(例如，标准Linux共享对象)，并且因此与使用这些标准分片对象的不同操作系统或平台兼容(例如，嵌入式Linux的各种Linux发行版和特色版(flavor))。

[0083] 在一些实施方式中，POSIX API 246提供用于与各种操作系统兼容的标准API。因此，语音助理客户端库240可以被包括在不同POSIX兼容操作系统的设备中，并且POSIX API 246提供语音助理客户端库240与不同操作系统之间的兼容性接口。

[0084] 在一些实施方式中，库包括支持和促进在实现语音助理(例如，定时器、警报、音量控制)的不同类型的设备上可用的基本用例的模块。

[0085] 在一些实施方式中，语音助理客户端库240包括控制器接口402，其包括用于启动、配置并且与语音助理交互的功能或模块。在一些实施方式中，控制器接口402包括用于在设备处启动语音助理的“开始()(Start())”功能或模块404；用于利用语音助理注册动作的“注册动作()(RegisterAction())”功能或模块406(例如，使得该动作是通过语音助理可动作的)，用于以更新的设置重新配置语音助理的“重新配置()(Reconfigure())”408功能，以及用于利用该助理注册用于基本事件的一组功能的“注册事件观察器()(RegisterEventObserver())”功能410。

[0086] 在一些实施方式中，语音助理客户端库240包括与特定语音助理功能相关联的多个功能或模块。例如，热词检测模块412处理语音输入以检测热词。话音处理模块414处理语音输入中的话音，并且将话音转换为文本或反之(例如，识别单词和短语，话音到文本数据转换，文本数据到话音转换)。动作处理模块416响应于口头输入来执行动作和操作。本地定时器/警报/音量控制模块418有助于设备处的警报闹钟、定时器和音量控制功能以及通过语音输入(例如，在设备处维护定时器、时钟、警报闹钟)对其进行控制。日志/度量模块420记录(例如日志)语音输入和响应，以及确定并记录相关度量(例如，响应时间、空闲时间等)。音频输入处理模块422处理语音输入的音频。MP3解码模块424解码MP3编码的音频。音频输入模块426通过音频输入设备(例如，麦克风)捕获音频。音频输出模块428通过音频输出设备(例如扬声器)输出音频。事件排队和状态跟踪模块430用于在设备处排队与语音助理相关联的事件并跟踪设备处的语音助理的状态。

[0087] 在一些实施方式中，调试库242提供用于调试的模块和功能。例如，HTTP服务器模块432促进调试连接问题，并且调试服务器/音频流模块434用于调试音频问题。

[0088] 在一些实施方式中，平台API 244提供语音助理客户端库240与设备的硬件功能之间的接口。例如，平台API包括用于捕获设备上的按钮输入的按钮输入接口436，用于捕获环

回音频的环回音频接口438，用于记录和确定度量的日志和度量接口440，用于捕获音频输入的音频输入接口442，用于输出音频的音频输出接口444以及用于使用可以与语音助理交互的其他服务来认证用户的认证接口446。图4所示的语音助理客户端库组织的优点在于，它可以在具有一致的API和语音助理功能集的各种语音助理设备类型上提供相同或类似的语音处理功能。此一致性支持语音助理应用的可移植性和语音助理操作的一致性，这继而又促进了一致的用户交互，以及对不同设备类型上执行的语音助理应用和功能的熟悉。在一些实施方式中，可以在服务器114处提供全部或部分语音助理客户端库240以支持基于服务器的语音助理应用(例如，对传送到服务器114进行处理的语音输入进行操作的服务器应用)。

[0089] 下面示出了对应于控制器402(“控制器”)和相关类的类和函数的示例代码。这些类和函数可以由在各种设备上可执行的应用经由通用API采用。

[0090] 下面的类“动作模块(ActionModule)”促进应用注册其自己的模块以处理由语音助理服务器提供的命令：

```
//应用可以注册自己的软件模块
//以处理语音助理服务器提供的命令。
class ActionModule {
    public:

//动作结果描述动作是否成功执行。
class Result{
    public:
        virtual~Result () = default;

//设置动作结果以指示成功。
virtual void SetOk () = 0;
```

[0091]

```
//将动作结果设置为给定的响应代码和人可读的串。
virtual void SetError (int response_code,
    const std:: string&str);
};

//对动作处理器的变元（Argument）。
class Args {
    public:
        virtual~Args () = 0;
//获取给定类型动作处理器(handler)变元的序列化 protobuf 数据。
virtual bool GetProtobufDataFromType
     (std:: string type, std:: string*
          data) = 0;
};
```

```
virtual~ActionModule () = 0;

//返回此模块的名称。
virtual std:: string GetName () = 0;

//用其| args |处理给定的| action_name |，并根据动作执行的成果更
//新结果。
virtual void Handle (std:: string action_name,
    std:: unique_ptr <Args> args, Result *result) = 0;

//将已命名的 protobuf 设置为给定的序列化数据以向语音助理指
//示此模块的本地状态。
virtual bool GetModuleContext (std:: string * protobuf_type,
    std:: string * protobuf_data) = 0;
};
```

[0093] 下面的类“构建信息(BuildInfo)”可用于描述运行语音助理客户端库240或语音助理客户端设备104本身的应用(例如,用该应用、平台和/或设备的标识符或版本号):

[0094]

```
//构建用于描述应用的信息
//运行语音助理客户端库。对于专用语音助理设备，这应该描述该
//设备。
//这个对象将从 CreateDefaultBuildInfo 返回，可以
//被修改，然后再设置回到设置对象。
class BuildInfo {
    public:
        virtual~BuildInfo () = default;

//设置应用版本。
```

```
        virtual void SetApplicationVersion (const std:: string&
                application_version) = 0;

    //设置安装标识符。这必须是特定于设备的标识符
    //其不应该与任何其他设备或用户标识符相同。
        virtual void SetInstallId (const std:: string&
                install_id) = 0;

    //设置平台标识符。
        virtual void SetPlatformId (const std:: string&
                platform_id) = 0;

    //设置平台版本。
        virtual void SetPlatformVersion (const std:: string&
                platform_version) = 0;

    //设置设备型号。可选的。
        virtual void SetDeviceModel (const std:: string&
                device_model) = 0;
    };
```

[0095]

[0096] 下面的类“事件委托(EventDelegate)”定义与基本事件相关联的函数，例如话音识别的开始，语音助理输出语音响应的开始和完成等：

```
    //从助理库接收事件。
    class EventDelegate {
        public:
            class RecognizedSpeechChangedEvent {
                public:
                    virtual~RecognizedSpeechChangedEvent () {}
```

[0097]

```
        // 指 示 来 自 语 音 助 理 的 更 新 的 识 别 文 本 。 如 果
    //OnRecognizedSpeechFinishedEvent 的部分，这表示最终的
        //识别文本。
                virtual std:: string
                    GetRecognizedSpeech () = 0;
            };

            virtual~EventDelegate () {}

        //指示语音助理客户端库正在启动。
            virtual void OnBootingUpEvent () = 0;

        //指示听到的是热词。
            virtual void OnHeardHotwordEvent () = 0;
```

[0098]

```
        //指示已经开始识别话音。话音识别将会
        //继续，直到收到 OnRecognizingSpeechFinishedEvent。
        virtual void OnRecognizingSpeechStartedEvent () = 0;

        //指示对已识别话音的当前假设的更改
        //发生。|event|指示新的假设。
            virtual void OnRecognizedSpeechChangedEvent (
                const RecognizedSpeechChangedEvent&event) = 0;

        //指示最终的话音识别已经发生。
        //| event |指示最终值。
            virtual void OnRecognizingSpeechFinishedEvent (
                const RecognizedSpeechChangedEvent&event) = 0;
```

```
//指示语音助理正在开始通过语音进行响应。
//语音助理将会响应，直到 OnRespondingFinishedEvent
//收到。
virtual void OnRespondingStartedEvent () = 0;

//指示语音助理已通过语音完成响应。
virtual void OnRespondingFinishedEvent () = 0;

//指示警报已开始播放。警报将继续
//播放直到收到 OnAlarmSoundingFinishedEvent。
virtual void OnAlarmSoundingStartedEvent () = 0;

//指示警报已完成播放。
virtual void OnAlarmSoundingFinishedEvent () = 0;
```

[0099]

```
//指示定时器已开始播放。定时器将继续
//播报直到收到 OnTimerSoundingFinishedEvent。
virtual void OnTimerSoundingStartedEvent () = 0;

//指示定时器已完成播放。
virtual void OnTimerSoundingFinishedEvent () = 0;

//指示已经对默认音量发生更改 (其
//例如在用户说“调大音量”时发生，无需
//指定警报或其他特定的音量类型。)| new_volume |
//指示从 0.0 到 1.0 的新默认音量。
virtual void OnDefaultVolumeChangeEvent (float
        new_volume) = 0;

//指示语音助理客户端库对于服务器已过时
```

```
        //需要更新。当发生这种情况时，客户端
        //不再与服务器交互
[0100]
        virtual void OnClientLibraryOutOfDateEvent () = 0;
        };
```

[0101] 下面的类“默认事件委托(DefaultEventDelegate)”定义了用于某些事件的不作为重写(do-nothing overrides)的函数：

```
        //提供 EventDelegate 的默认不作为实施方式，
        //对于只重写那些引起兴趣的函数是有用的。
        class DefaultEventDelegate: public EventDelegate {
            public:
            void OnBootingUpEvent () override {}
            void OnHeardHotwordEvent () override {}
            void OnRecognizingSpeechStartedEvent () override {}
            void OnRecognizedSpeechChangedEvent (const
                RecognizedSpeechChangedEvent&event)override {}
            void OnRecognizingSpeechFinishedEvent (const
[0102]          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 {}
        };
```

[0103] 下面的类“设置(Settings)”定义了可以被提供给控制器402的设置(例如,场所、地理位置、文件系统目录)。

```
//提供给控制器的助理设置。它们必须是
//在启动助理时提供给控制器。它们也可以
//被更新，然后提供给 Reconfigure 函数以生效。
//嵌入的应用程序不应该创建由此衍生的自己的类。
class Settings {
    public:
        virtual~Settings () {}

//创建默认的 BuildInfo 对象。
virtual std:: unique_ptr <BuildInfo>
    CreateDefaultBuildInfo () = 0;

//设置设备的地理位置。可选的。
```

[0104]

```
virtual void SetGeolocation (const Geolocation&
    geolocation) = 0;

//设置设备的编译信息。
virtual void SetBuildInfo (const BuildInfo&
    build_info) = 0;

//设置语音助理客户端库可以使用的文件系统目录。
//每当语音助理客户端库丢失所有以前的上下文时都应该清除该目录
//例如当工厂数据
//发生重置时。
virtual void SetAssistantDirectory (const
    std:: string&path) = 0;
```

```
//将 UserAgent 设置为连接到服务器。
virtual void SetUserAgent (const std:: string&
    user_agent) = 0;
```

[0105]

```
//设置设备的场所。
virtual void SetLocaleInfo (const LocaleInfo&
    locale_info) = 0;
};
```

[0106] 下面的类“控制器(Controller)”对应于控制器402,并且Start(),Reconfigure(),RegisterAction()和RegisterEventObserver()函数分别对应于函数Start()404,Reconfigure()408,RegisterAction()406和RegisterEventObserver()410。

```
//助理的控制器类。
class Controller {
    public:
        virtual~Controller () {}

//创建应用应该配置的新默认设置对象
//然后传递给 Start。
```

[0107]

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

//启动助理并立即返回。成功时返回真，
//失败时为假。每个进程只会成功一次。|settings|是
//助理模块的设置。这些通过 const 引用传递
//所以很明显，调用者保留设置对象并且
//除非传递给 Reconfigure，否则任何后续更改无效。
```

```
//如果没有设置任何必需的设置，此函数将失败。
virtual bool Start (const Settings & settings) = 0;

//重新配置运行助理并立即返回。
//在包括助理尚未启动的失败时，返回假。
//|settings|是语音助理模块的新设置。
//如果没有设置任何必需的设置，此函数将失败。
virtual bool Reconfigure (const Settings & settings) = 0;

//注册动作|module|。如果已经注册，则失败。
virtual bool RegisterAction (std:: unique_ptr <ActionModule>)
```

[0108]

```
    module) = 0;

//注册 EventDelegate 以接收所有助理事件。
virtual void RegisterEventObserver (
    std:: unique_ptr <EventDelegate> delegate) = 0;

//调用这个函数来创建控制助理的控制器类。
//|platform|必须设置为指向平台 API 的指针
//助理将使用它。错误时返回 nullptr。
static ASSISTANT_EXPORT std:: unique_ptr <Controller>
    Create (std:: unique_ptr <PlatformApi> platform_api);
};
```

[0109] 在一些实施方式中，语音助理客户端设备104或者投射设备106实现平台(例如，用于使用相同平台与其他设备进行通信的接口集以及被配置为支持该接口集的操作系统)。下面的示例代码示出了与语音助理客户端库402同平台进行交互的接口相关联的函数。

[0110] 下面的类“认证(Authentication)”定义了用于以特定账户认证语音助理的用户的认证令牌：

```
//平台的认证提供者。
class Authentication {
public:

//返回认证令牌的认证范围。
virtual std:: string GetGoogleOAuth2Scopes () = 0;
```

[0111]

```
//返回认证令牌。
virtual bool GetGoogleOAuth2Token (std:: string *
    token) = 0;

protected:
    virtual~Authentication () = default;
};
```

[0112] 下面的类“输出流类型(OutputStreamType)”定义了音频输出流的类型：

```
//可能的音频输出流类型。
enum OutputStreamType {
    KTts,
```

[0113]

```
    kAlarm,
    kCalibration,
};
```

[0114] 下面的类“样本格式(SampleFormat)”定义了支持的音频样本格式(例如,PCM格式)：

```
//支持的 PCM 样本格式。
enum SampleFormat {
    kInterleavedS16，//交错的带符号的 16 位整数。
    kInterleavedS32，//交错的带符号的 32 位整数。
    kInterleavedF32，//交错的 32 位浮点数。
    kPlanarS16，//平面的带符号的 16 位整数。
    kPlanarS32，//平面的带符号的 32 位整数。
    kPlanarF32，//平面的 32 位浮点数。
};
```

[0116] 下面的“缓冲器格式(BufferFormat)”定义了存储在设备处的音频缓冲器中的数据的格式：

```
//有关存储在音频缓冲器中的数据格式的信息。
struct BufferFormat {
    int sample_rate;
    SampleFormat sample_format;
    int num_channels;
};
```

[0117]

[0118] 类“音频缓冲器(AudioBuffer)”定义用于音频数据的缓冲器：

```
//用于输入/输出音频数据的缓冲器类。
class AudioBuffer {
    public:

//返回缓冲器中数据的格式。
    virtual BufferFormat GetFormat () const = 0;

//不可变的数据；由 AudioInput 委托使用来读取传入
```

[0119]

```
//数据。
    virtual const char * GetData () const = 0;

//可写入的数据；由 AudioOutput 委托使用以写入更多
//输出数据。
virtual char * GetWritableData () const = 0;
```

[0120]

```
//返回包含在
//GetData ()/GetWritableData ()中的音频帧数量。
    virtual int GetFrames () const = 0;

 protected:
    virtual~AudioBuffer () {}
};
```

[0121] 下面的类“音频输出(AudioOutput)”定义了用于音频输出的接口：

```
//用于音频输出的接口。
class AudioOutput {
    public:
        enum Error{
            kFatalError,
            kUnderrun,
```

[0122]

```
};

class Delegate{
public:

//需要更多输出音频数据时调用。委托
//实施方式必须尽快将数据填充到| buffer |中，
```

```
//一旦有数据被写入则调用| done_cb |。注意
//委托可以部分填充缓冲器，但是
//| bytes_written |数量必须是帧大小的倍数。委托
//不会取得| buffer |的所有权。
//注意这个方法不能被阻断。如果没有可用的数据
//立即填充缓冲器，可以
//由任何线程异步填充缓冲器，然后必须调用| done_cb |。
//| done_cb |在通过调用 Stop ()停止流后不能调用。
//如果已经到达流的结束，那么
//委托必须用 0 | bytes_written |调用| done_cb |。
virtual void FillBuffer (AudioBuffer * buffer,
    const std:: function <void (int
    frames_written)>&done_cb) = 0;
```

[0123]

```
//调用来表示流的结束 (即，
//将 0 | bytes_written |传递到 FillBuffer () 的| done_cb |的委托的点)
//被播放。一旦这个被调用，可以安全地调用 Stop ()
//而没有任何未播放数据的风险。
virtual void OnEndOfStream () = 0;
//发生输出错误时调用。
virtual void OnError (Error error) = 0;

//一旦输出停止，则调用。一旦这个方法被
//调用，将不会再有对任何委托方法的调用，除非
//输出再次开始。
virtual void OnStopped () = 0;

protected:
~Delegate () {}
};
```

```
virtual~AudioOutput () {}

//返回此输出的流类型，该流类型是在
//输出创建时指定的。
virtual OutputStreamType GetType () = 0;

//开始音频输出。这将以给定|format|
//通过调用| delegate |的 FillBuffer () 方法开始请求缓冲器。
virtual void Start (const BufferFormat& format,
Delegate* delegate) = 0;

//停止音频输出，将此接口置于 Start ()可以
//以新的音频格式安全地再次调用并委托的状态。
//调用 Stop()时，应该放弃由委托提供的
//任何未播放数据。
//一旦停止完成并且不再有对该委托进行调用，
//委托的 OnStopped () 方法将被调用。
virtual void Stop () = 0;

//设置此输出流的音量范围。
//只要该音量在| min_volume | <=音量<= | max_volume |范围内，
//此流的音量就应跟踪默认音量(所以，使用默认
//音量，但是被限制在给定的范围内)。| min_volume |和
//| max_volume |值为 0.0 <= v <= 1.0，并且表示
//系统的总可能输出音量的一部分。
virtual void SetVolume (float min_volume, float
max_volume) = 0;
};
```

[0124]

[0125] 下面的类“音频输入(AudioInput)”定义了用于捕获音频输入的接口：

```
//用于捕获音频输入的接口。开始时，这应该捕获
//来自所有麦克风的音频，并将每个麦克风的数据提供为
//在提供给委托的
//OnBufferAvailable () 方法的缓冲器中的单独通道。
class AudioInput {
    public:
        enum Error {
            kFatalError,
            kOverrun,
};

class Delegate{
    public:
//有更多输入音频数据可用时调用。|timestamp|是
//以微秒为单位的时间 (相对于 CLOCK_MONOTONIC_RAW 时
期)
//在| buffer |中的数据被捕获 (对于环回音频，它是
//预计数据被播放时的时间戳)。
virtual void OnBufferAvailable (
    const AudioBuffer& buffer,
    int64_t timestamp) = 0;

//当 AudioInput 发生错误时调用。
virtual void OnError (Error error) = 0;

//一旦输入停止，则调用。一旦这个方法被
//调用，将不会再有对任何委托方法的调用，除非
//输入再次开始。
```

[0126]

```
virtual void OnStopped () = 0;
};

virtual~AudioInput () {}

//开始捕获音频输入并将其传递给| delegate |的
//OnBufferAvailable ()方法。
```

[0127]

```
virtual void Start (Delegate * delegate) = 0;

//停止捕获音频输入。一旦输入停止并且对
//任何委托方法不再进行调用，委托的 OnStopped ()方法
//将被调用。
virtual void Stop () = 0;
};
```

[0128] 下面的类“资源(Resources)”定义对系统资源的访问：

```
//访问系统资源文件。
class Resources {
    public:
        using ResourceLoadingCallback = std::function<void(
            const std::string& output)>;
        Resources() {}
```

[0129]

```
        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;
    };
```

[0130]

[0131] 下面的类“平台Api(PlatformApi)”为语音助理客户端库240指定平台API(例如，平台API 244)：

```
    //使用语音助理的平台 API。
    class PlatformApi {
    public:
    virtual~PlatformApi () {}
```

[0132]

```
    //返回用于流的所需| type |的音频输出接口。
    //这由 PlatformApi 拥有。
    virtual std:: unique_ptr <AudioOutput> GetAudioOutput (
        OutputStreamTypetype) = 0;

    //返回用于捕获音频输入的接口。
```

```
virtual std:: unique_ptr <AudioInput> GetAudioInput () = 0;

//返回用于捕获环回音频的接口。这是
//所捕获的数据是即将播放的音频数据的
//“音频输入”。
//所有混音和后处理已完成后，
[0133] //在发送到输出硬件前尽快地，
//都可以捕获环回音频。
virtual std:: unique_ptr <AudioInput>
    GetLoopbackInput () = 0;

virtual Authentication& GetAuthentication () = 0;
};
```

[0134] 在一些实施方式中，可以在语音助理客户端库240之外处理音量控制。例如，系统音量可以由设备在语音助理客户端库240的控制之外维持。作为另一示例，语音助理客户端库240仍然可以支持音量控制，但是对语音助理客户端库240的音量控制请求被定向到设备。

[0135] 在一些实施方式中，语音助理客户端库240中的警报和定时器功能可以由用户禁用或当在设备处实现库时被禁用。

[0136] 在一些实施方式中，语音助理客户端库240还支持到设备上的LED的接口，以促进在设备LED上显示LED动画。

[0137] 在一些实施方式中，语音助理客户端库240可以被包括在投射设备106处的投射接收器模块(例如，接收器模块146)中或由其链接。语音助理客户端库240与接收器模块146之间的链接可以包括例如对附加动作(例如，本地媒体播放)的支持，以及对投射设备106上的LED的控制的支持。

[0138] 图5示出了根据一些实施方式的用于处理设备上的口头输入的方法500的流程图。方法500在具有音频输入系统(例如音频输入设备108/132)，一个或多个处理器(例如，处理单元202)以及存储由一个或多个处理器执行的一个或多个程序的存储器(例如，存储器206)的电子设备(例如，语音助理客户端设备104，投射设备106)处执行。在一些实施方式中，电子设备包括音频输入系统(例如音频输入设备108/132)，一个或多个处理器(例如，处理单元202)和存储由一个或多个处理器执行的一个或多个程序的存储器(例如，存储器206)，一个或多个程序包括用于执行方法500的指令。在一些实施方式中，非暂时性计算机可读存储介质存储一个或多个程序，一个或多个程序包括当由具有音频输入系统(例如，音频输入设备108/132)和一个或多个处理器(例如，处理单元202)的电子设备执行时，使电子设备执行方法500的指令。用于执行方法500的程序或指令可以被包括在以上参照图2-4描

述的模块、库等中。

[0139] 设备接收(502)设备处的口头输入。客户端设备104/投射设备106捕获由用户发出的口头输入(例如,语音输入)。

[0140] 设备处理(504)口头输入。客户端设备104/投射设备106处理口头输入。该处理可以包括热词检测,到文本数据的转换以及与用户提供的命令、请求和/或参数相对应的单词和短语的标识。在一些实施方式中,处理可以是最小的,或者可以根本没有处理。例如,处理可以包括对口头输入音频进行编码以便传输到服务器114,或者准备所捕获的口头输入的原始音频以便传输到服务器114。

[0141] 设备向远程系统传送(506)请求,该请求包括基于口头输入确定的信息。客户端设备104/投射设备106通过处理口头输入来确定来自口头输入的请求,以识别来自口头输入的请求和一个或多个相关参数。客户端设备104/投射设备106将所确定的请求传送到远程系统(例如,服务器114),其中远程系统确定并产生对该请求的响应。在一些实施方式中,客户端设备104/投射设备106将语音输入(例如,作为编码音频、作为原始音频数据)传送到服务器114,并且服务器114处理口头输入以确定请求和相关参数。

[0142] 设备接收(508)对请求的响应,其中响应是由远程系统根据基于口头输入的信息生成的。远程系统(例如,服务器114)确定并生成对该请求的响应,并将该响应传送到客户端设备104/投射设备106。

[0143] 设备根据响应执行(510)操作。客户端设备104/投射设备106根据接收到的响应执行一个或多个操作。例如,如果响应是通过音频向设备输出某些信息的命令,则客户端设备104/投射设备106检索信息,将信息转换为话音音频输出,并且通过扬声器输出话音音频。作为另一示例,如果响应是设备播放媒体内容的命令,则客户端设备104/投射设备106检索媒体内容并播放媒体内容。

[0144] 通过在电子设备上执行的语音助理库的一个或多个语音处理模块执行接收、处理、传送、接收和执行中的一个或多个,语音处理模块提供多个语音处理操作,所述语音处理操作是在电子设备(512)上执行或可执行的一个或多个应用程序和/或操作软件可访问的。客户端设备104/投射设备106可以具有语音助理客户端库240,其包括用于执行接收、处理、传送、接收和执行步骤中的一个或多个的功能和模块。语音助理客户端库240的模块提供多个语音处理和助理操作,所述语音处理操作是在包括或链接到库240的客户端设备104/投射设备106处的应用、操作系统和平台软件处可访问的(例如,运行库240和相关的API)。

[0145] 在一些实施方式中,在远程系统上执行与语音处理模块相关联的至少一些语音处理操作,该远程系统经由广域网与电子设备互连。例如,用于确定请求的口头输入的处理可以由服务器114执行,服务器114通过网络112与客户端设备104/投射设备106连接。

[0146] 在一些实施方式中,语音助理库是在多个不同设备类型上可操作的公共操作系统上可执行的,由此实现被配置为与语音处理操作中的一个或多个交互的语音使能应用的可移植性。语音助理客户端库240(以及相关库和API,例如调试库242、平台API 244、POSIX API 246)使用预定义操作系统(例如,Linux)的标准元件(例如,对象),并且因此在运行预定义操作系统的发行版或特色版的各种设备上(例如,不同的Linux或基于Linux的发行版或特色版)上是可操作的。以这种方式,语音助理功能可用于各种设备,并且语音助理体验

在各种设备中是一致的。

[0147] 在一些实施方式中，可以在设备处处理请求和响应。例如，对于设备本地的基本功能，例如定时器、警报闹钟、时钟和音量控制，客户端设备104/投射设备106可以处理口头输入并且确定该请求对应于这些基本功能中的一个，确定设备处的响应，并根据响应执行一个或多个操作。为了记录目的，设备仍然可以向服务器114报告请求和响应。

[0148] 在一些实施方式中，用于包括音频输入系统的电子设备的设备不可知语音助理库包括一个或多个语音处理模块，该一个或多个语音处理模块被配置为在多个不同电子设备类型上实现的公共操作系统上执行，语音处理模块提供可在电子设备上执行的应用程序和操作软件访问的多个语音处理操作，从而实现被配置为与一个或多个语音处理操作交互的语音使能应用的可移植性。语音助理客户端库240是可以在与库共享相同的预定义操作系统基础的各种设备上运行的库(例如，库和设备操作系统是基于Linux的)，因此库是设备不可知。库240提供用于在各种设备上的应用可访问的语音助理功能的多个模块。

[0149] 在一些实施方式中，与语音处理模块相关联的至少一些语音处理操作在经由广域网与电子设备互连的后端服务器上执行。例如，库240包括与服务器114通信以将口头输入传送到服务器114以进行处理来确定请求的模块。

[0150] 在一些实施方式中，语音处理操作包括被配置为控制与电子设备耦合(例如，直接或通信地)的设备的特定于设备的操作。库240可以包括用于控制耦合到客户端设备104/投射设备106的其他设备(例如无线扬声器、智能电视等)的功能或模块。

[0151] 在一些实施方式中，语音处理操作包括信息和媒体请求操作，其被配置为向电子设备的用户或在与电子设备耦合(例如，直接或通信地)的设备上的用户提供所请求的信息和/或媒体内容。库240可以包括用于检索信息或媒体并且在客户端设备104/投射设备106上或在耦合的设备上提供信息或媒体(例如，大声读出电子邮件、大声朗读新闻文章、播放流式音乐)的功能或模块。

[0152] 应该理解，虽然术语“第一”、“第二”等可以在本文用于描述各种元件，但是这些元件不应该受这些术语的限制。这些术语仅用于区分一个元件与和另一个元件。例如，第一接触可以被称为第二接触，并且类似地，第二接触可以被称为第一接触，这更改了描述的含义，只要所有发生的“第一接触”被一致地重命名并且所有发生的第二接触都一致地重命名。第一接触和第二接触都是接触，但它们不是同一接触。

[0153] 本文使用的术语仅用于描述特定实施方式的目的，而不意图限制权利要求。如在实施方式和所附权利要求的描述中所使用的，除非上下文另外清楚地指出，否则单数形式“一”、“一个”和“该”旨在也包括复数形式。还将理解的是，本文所使用的术语“和/或”是指并且包含一个或多个相关所列项目的任何和所有可能的组合。将进一步理解的是，当在本说明书中使用时，术语“包括”和/或“包含”指定所陈述的特征、整体、步骤、操作、元件和/或组件的存在，但不排除存在或添加一个或多个其他特征、整体、步骤、操作、元件、组件和/或其组合。

[0154] 如本文所使用的，术语“如果”根据上下文可被解释为意指“当”或“在…时”或“响应于确定”或“根据确定”或“响应于检测”到先决条件为真。类似地，短语“如果确定[所陈述的先决条件为真]”或“如果[所陈述的先决条件为真]”或“当[所陈述的先决条件为真”]根据上下文可以被解释为“在确定时”或“响应于确定”或“根据确定”或“根据检测”或“响应于

检测”到所陈述的先决条件为真。

[0155] 现在将详细参照各种实施方式，其示例在附图中示出。在以下详细描述中，阐述了许多具体细节以提供对本发明和所描述的实施方式的透彻理解。然而，可以在没有这些具体细节的情况下实施本发明。在其他实例中，众所周知的方法、过程、组件和电路未被详细描述，以免不必要地模糊实施方式的方面。

[0156] 为了解释的目的，前面的描述已经参照具体实施方式进行了描述。然而，以上的说明性讨论并非旨在穷举或将本发明限制于所公开的确切形式。鉴于上述教导，许多修改和变化是可能的。选择和描述实施方式是为了最好地解释本发明的原理及其实际应用，从而使本领域的其他技术人员能够最佳地利用本发明以及具有适合于预期的特定用途的各种修改的各种实施方式。

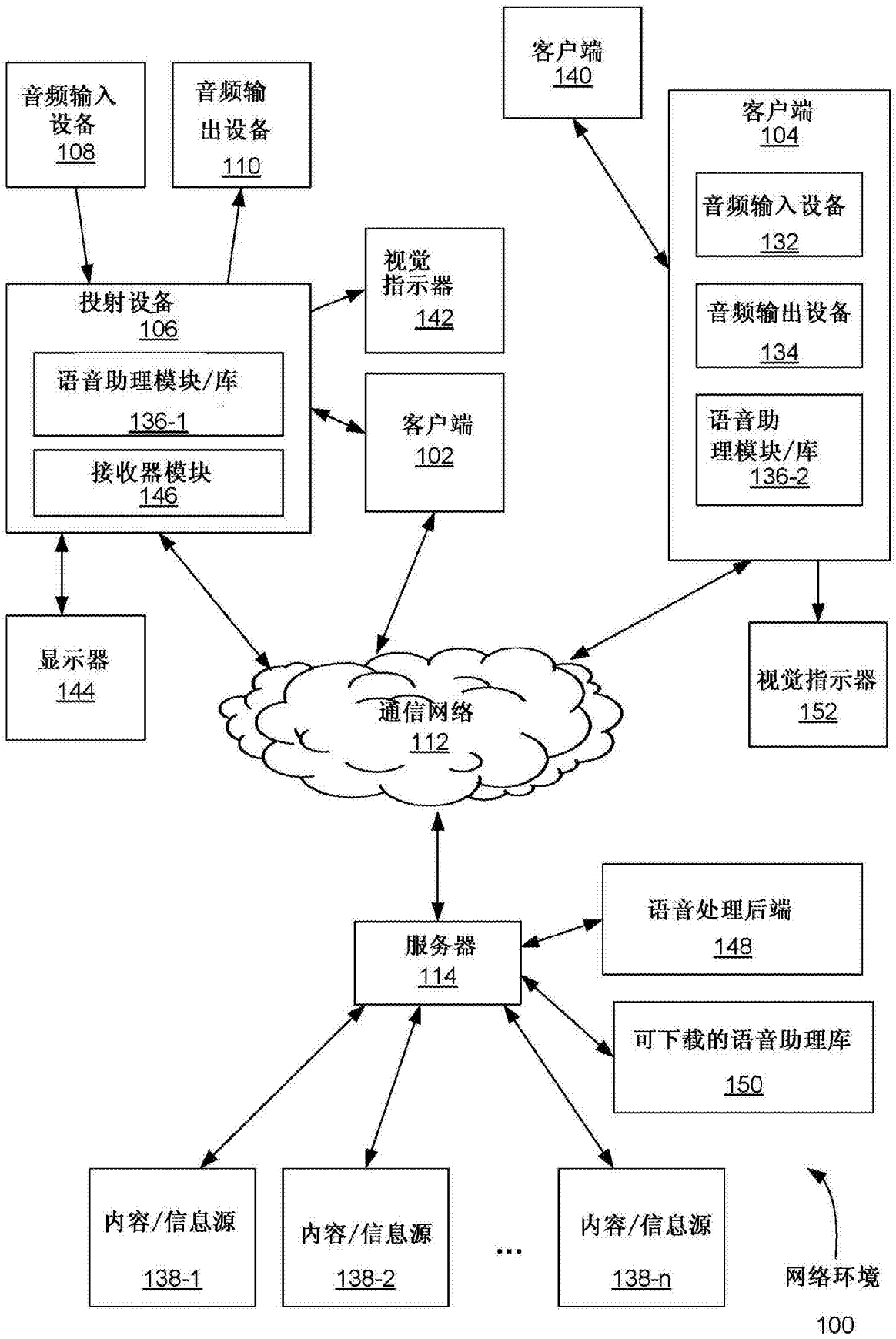

客户端
140
音频输入
设备
108
音频输
出设备
110
客户端
104
音频输入设备
132
视觉
指示器
142
音频输出设备
134
投射设备
106
语音助理模块/库
136-1
客户端
102
语音助
理模块/库
136-2
接收器模块
146
显示器
144
通信网络
112
视觉指示器
152
语音处理后端
148
服务器
114
可下载的语音助理库
150
内容/信息源
138-1
内容/信息源
138-2
...
内容/信息源
138-n
网络环境
100

图1

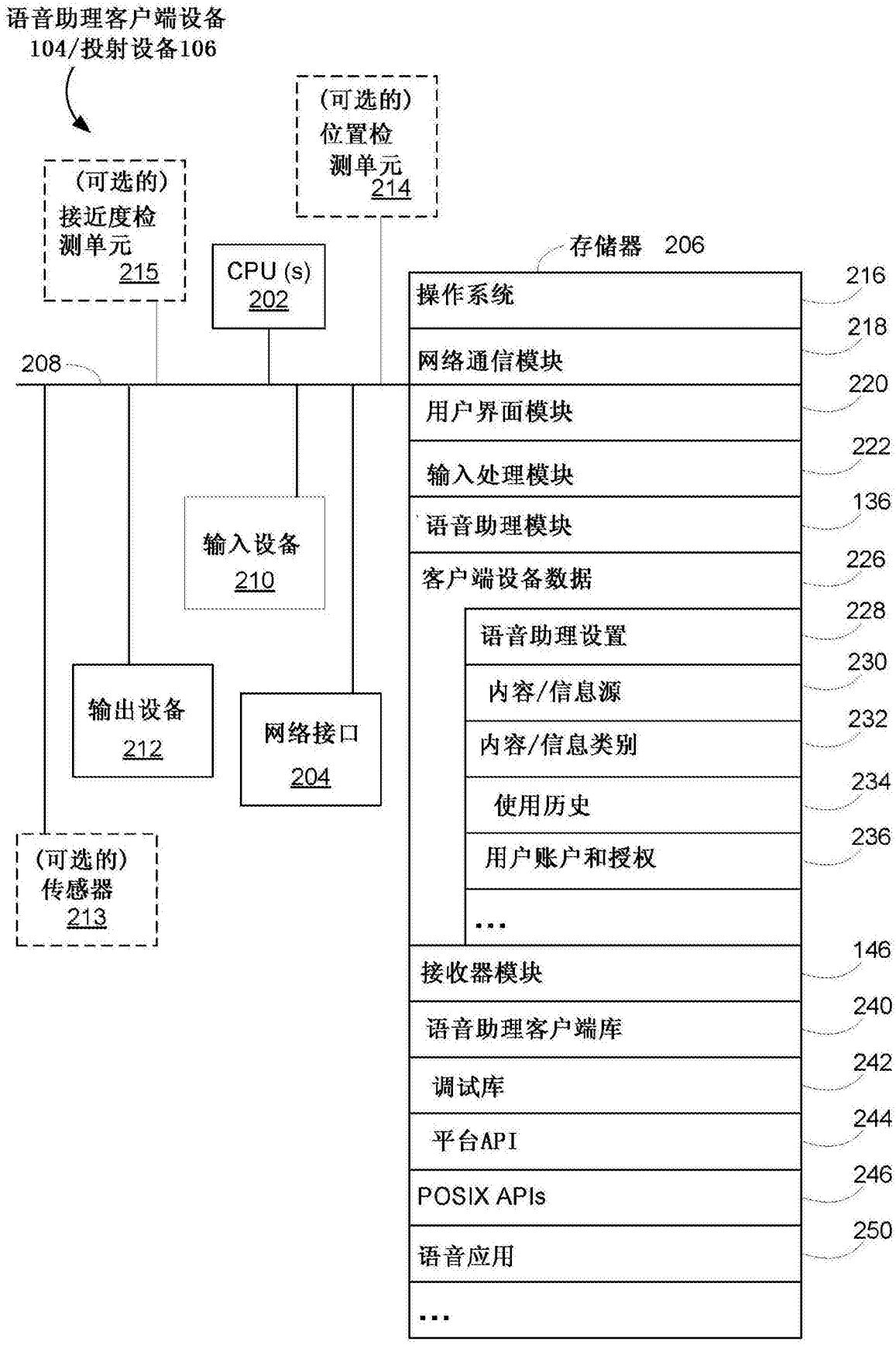

语音助理客户端设备
104/投射设备106
(可选的)
位置检
测单元
214
(可选的)
接近度检
测单元
215
CPU (s)
202
存储器 206
操作系统
216
网络通信模块
218
208
用户界面模块
220
输入处理模块
222
语音助理模块
136
输入设备
210
客户端设备数据
226
语音助理设置
228
内容/信息源
230
内容/信息类别
232
输出设备
212
网络接口
204
使用历史
234
用户账户和授权
236
(可选的)
传感器
213
...
接收器模块
146
语音助理客户端库
240
调试库
242
平台API
244
POSIX APIs
246
语音应用
250
...

图2

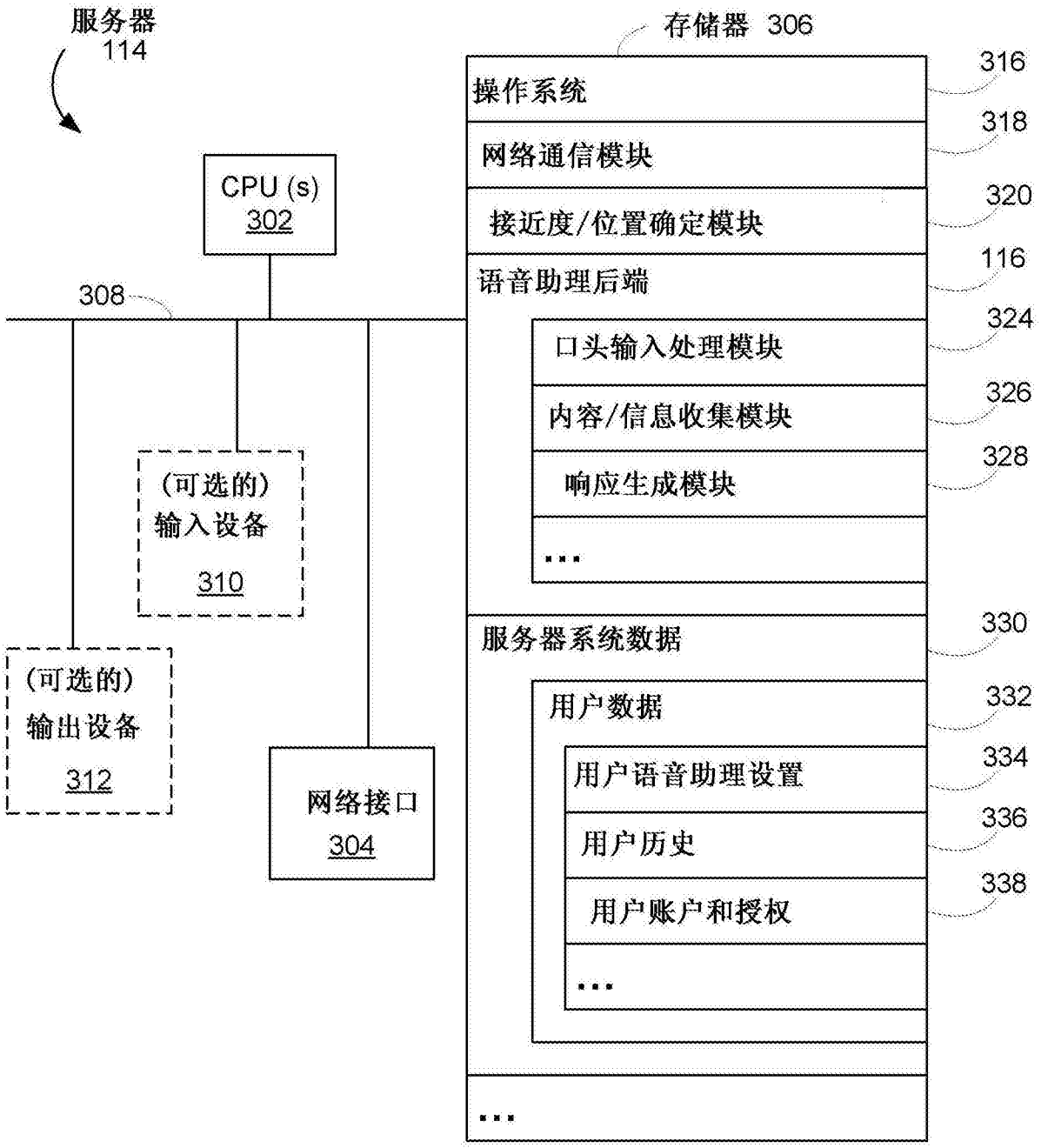

服务器
114
CPU (s)
302
308
(可选的)
输入设备
310
(可选的)
输出设备
312
网络接口
304
存储器 306
操作系统
316
网络通信模块
318
接近度/位置确定模块
320
语音助理后端
116
口头输入处理模块
324
内容/信息收集模块
326
响应生成模块
328
...
服务器系统数据
330
用户数据
332
用户语音助理设置
334
用户历史
336
用户账户和授权
338
...
...

图3

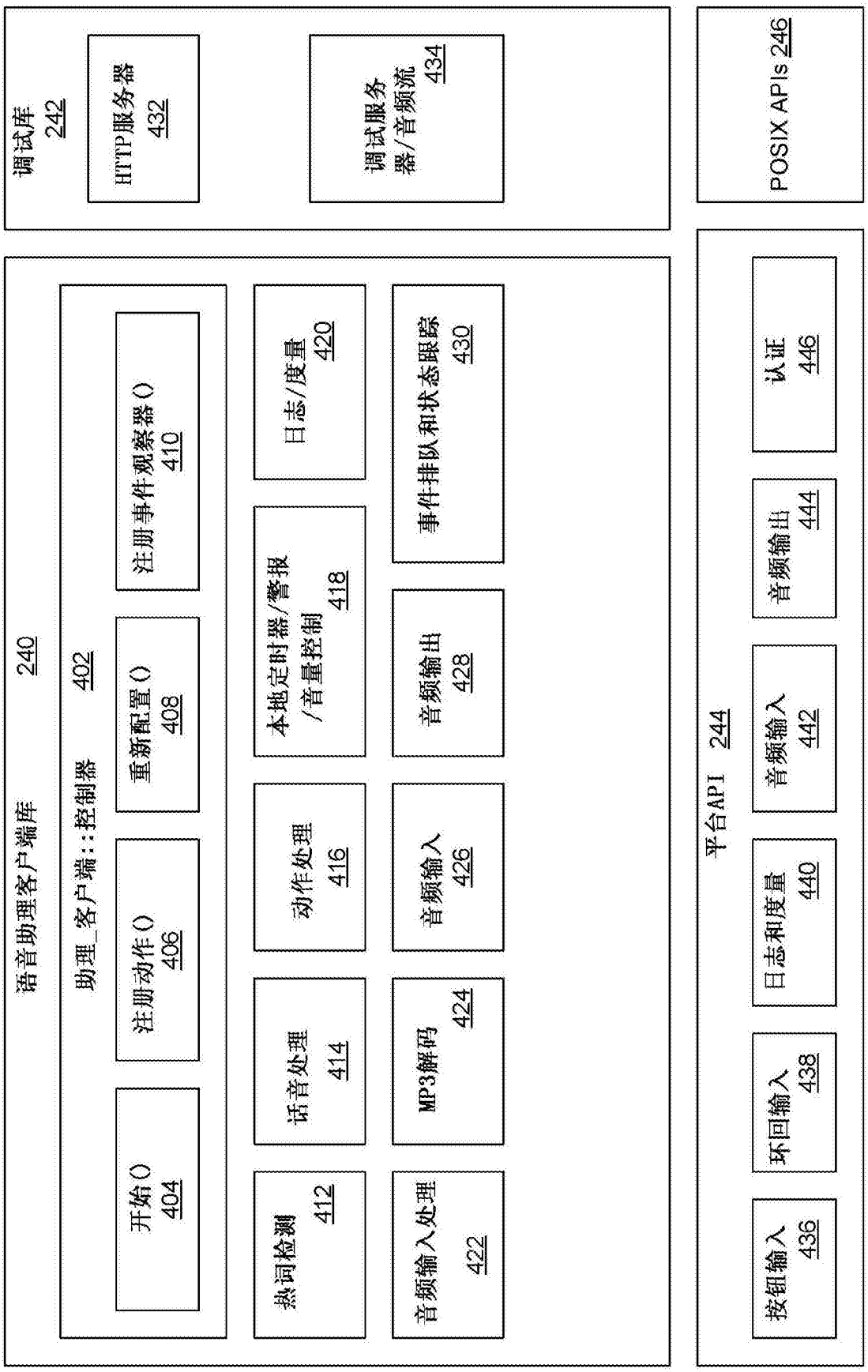

调试库
242
HTTP服务器
432
调试服务器/音频流
434
POSIX APIs 246
语音助理客户端库
240
助理_客户端::控制器
402
开始()
404
注册动作()
406
重新配置()
408
注册事件观察器()
410
热词检测
412
话音处理
414
动作处理
416
本地定时器/警报/音量控制
418
日志/度量
420
音频输入处理
422
MP3解码
424
音频输入
426
音频输出
428
事件排队和状态跟踪
430
平台API 244
按钮输入
436
环回输入
438
日志和度量
440
音频输入
442
音频输出
444
认证
446

图4

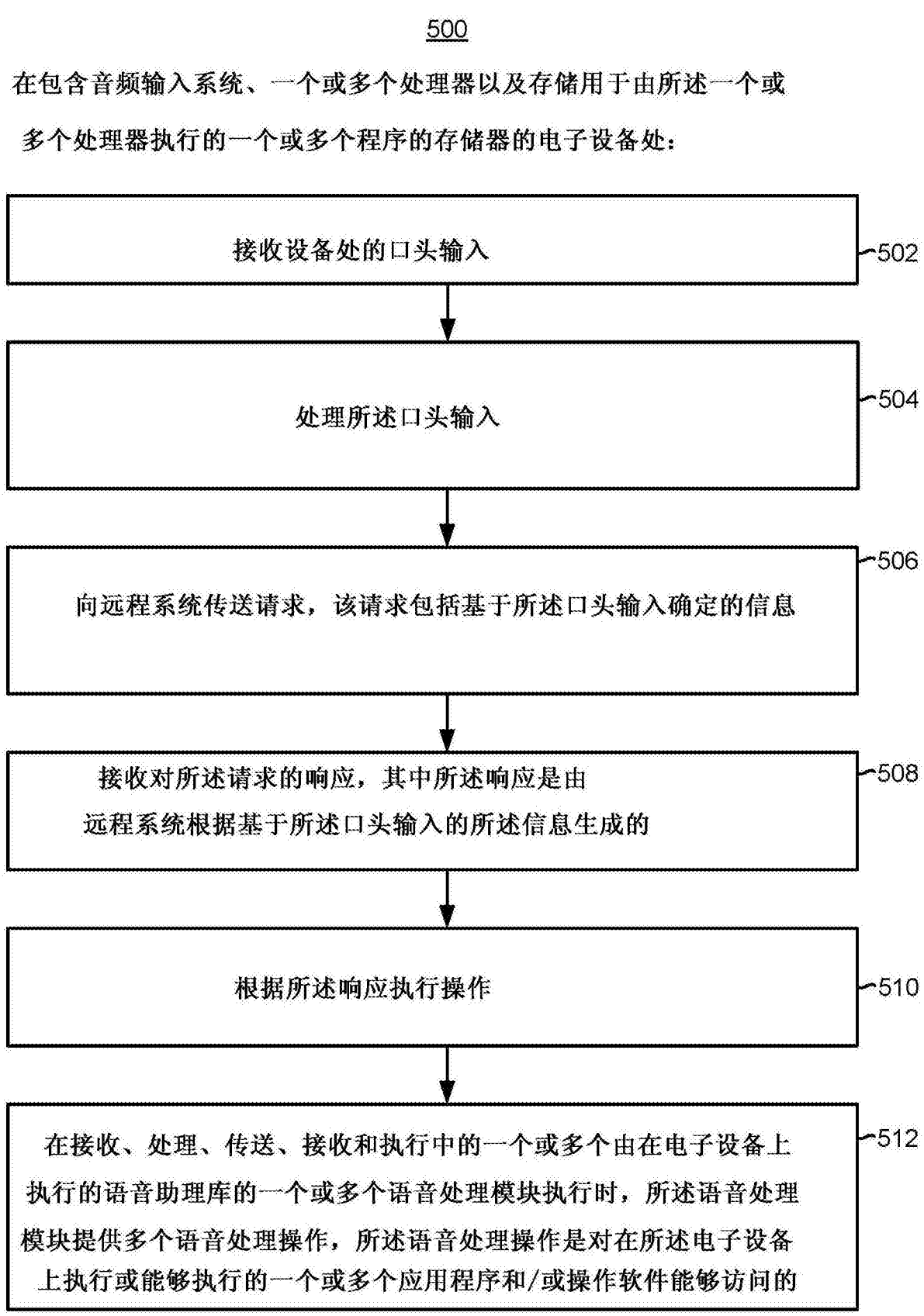

500
在包含音频输入系统、一个或多个处理器以及存储用于由所述一个或多个处理器执行的一个或多个程序的存储器的电子设备处：
接收设备处的口头输入
502
处理所述口头输入
504
向远程系统传送请求，该请求包括基于所述口头输入确定的信息
506
接收对所述请求的响应，其中所述响应是由远程系统根据基于所述口头输入的所述信息生成的
508
根据所述响应执行操作
510
在接收、处理、传送、接收和执行中的一个或多个由在电子设备上执行的语音助理库的一个或多个语音处理模块执行时，所述语音处理模块提供多个语音处理操作，所述语音处理操作是对在所述电子设备上执行或能够执行的一个或多个应用程序和/或操作软件能够访问的
512

图5