(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 112328330 A
(43)申请公布日 2021.02.05

(21)申请号 202011341208.1

(22)申请日 2020.11.25

(71)申请人 北京五八信息技术有限公司
地址 100083 北京市海淀区学清路甲18号中关村东升科技园学院园三层301室

(72)发明人 桑锐

(74)专利代理机构 北京润泽恒知识产权代理有限公司 11319
代理人 吕俊秀

(51)Int.Cl.
G06F 9/445(2018.01)
G06F 16/955(2019.01)

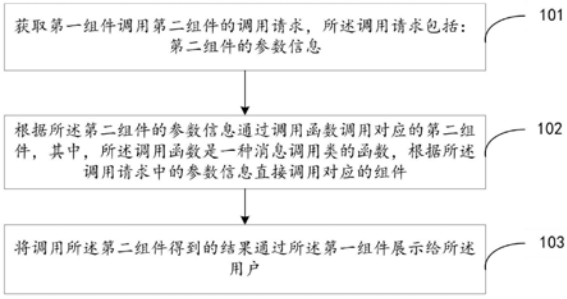
权利要求书2页 说明书10页 附图4页

(54)发明名称

iOS组件的调用方法、装置、中间件、电子设备及介质

(57)摘要

本发明提供一种iOS终端的组件调用方法，装置、中间件、电子设备存储介质，所述方法包括：获取第一组件调用第二组件的调用请求时，根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件；并将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。也就会说，本发明实施例中，通过中间件直接调用对应的组件进行信息传递，增加了各组件的独立性，降低了组件间的相互关联，消除组件间耦合以及组件和中间件的耦合性，降低了学习成本，且保证了安全行可控。



CN 112328330 A

1.一种iOS终端的组件调用方法,其特征在于,包括:

获取第一组件调用第二组件的调用请求,所述调用请求包括:第二组件的参数信息;

根据所述第二组件的参数信息通过调用函数调用对应的第二组件,其中,所述调用函数是一种消息调用类的函数,根据所述调用请求中的参数信息直接调用对应的组件;

将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。

2.根据权利要求1所述的iOS终端的组件调用方法,其特征在于,所述根据所述第二组件的参数信息通过调用函数调用对应的第二组件,包括:

根据所述第二组件的参数信息读取对应文件lib包的头文件;

获取所述头文件对应组件的含义以及应用程序接口API的调用方法;

根据所述组件的含义以及API的调用方法通过调用函数调用对应的第二组件,并在所述组件调用中删除中间件的硬代码和分类文。

3.根据权利要求1或2所述的iOS终端的组件调用方法,其特征在于,在获取所述第一组件调用第二组件的调用请求之前,所述方法还包括:

通过移动设备操作系统运行时iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式,其中,所述组件至少包括第一组件和第二组件;

将所有组件中的每个组件的源码封装成对应的文件lib包;

显示每个lib包的头文件,以便于根据所述头文件获取对应组件的含义以及应用程序接口API的调用方法。

4.根据权利要求1或2所述的iOS终端的组件调用方法,其特征在于,所述方法还包括:

接收到外部调用应用程序APP;

将所述调用APP的统一资源定位标识符URL映射成对应的类class进行调起。

5.一种iOS终端的组件调用装置,其特征在于,包括:

第一获取模块,用于获取第一组件调用第二组件的调用请求,所述调用请求包括:第二组件的参数信息;

调用模块,用于根据所述第二组件的参数信息通过调用函数调用对应的第二组件,其中,所述调用函数是一种消息调用类的函数,根据所述调用请求中的参数信息直接调用对应的组件;

第一显示模块,用于将调用所述第二组件得到的结果通过所述第一组件显示给所述用户。

6.根据权利要求5所述的iOS终端的组件调用装置,其特征在于,所述调用模块包括:

读取模块,用于根据所述第二组件的参数信息读取对应文件lib包的头文件;

第二获取模块,用于获取所述头文件对应组件的含义以及应用程序接口API的调用方法;

调用子模块,用于根据所述组件的含义以及API的调用方法通过调用函数调用对应的第二组件,并在调用中删除中间件的硬代码和分类文。

7.根据权利要求5或6所述的iOS终端的组件调用装置,其特征在于,所述装置还包括:

创建模块,用于在所述第一获取模块获取所述第一组件调用第二组件的调用请求之前,通过移动设备操作系统运行时iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式,其中,所述组件至少包括第一组件和第二组件;

封装模块,用于将所有组件中的每个组件的源码封装成对应的文件lib包;

第二显示模块,用于显示每个lib包的头文件,以便于根据所述头文件获取对应组件的含义以及应用程序接口API的调用方法。

8.根据权利要求5或6所述的iOS终端的组件调用装置,其特征在于,所述装置还包括:

接收模块,用于接收到外部调用应用程序App;

映射模块,用于将所述调用App的统一资源定位标识符URL映射成对应的类class进行调起。

9.一种基于iOS终端的组件调用的中间件,其特征在于,包括:

移动设备操作系统运行时iOS runtime模块,用于动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式;

调用函数模块,用于获取第一组件调用第二组件的调用请求,所述调用请求包括:第二组件的参数信息,根据所述第二组件的参数信息通过调用函数调用对应的第二组件,其中,所述调用函数是一种消息调用类的函数,根据所述调用请求中的参数信息直接调用对应的组件;将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。

10.根据权利要求9所述的基于iOS终端的组件调用的中间件,其特征在于,

所述调用函数模块,还用于在根据所述第二组件的参数信息调用对应的第二组件中删除所述中间件的硬代码和分类文。

11.一种电子设备,其特征在于,包括:

处理器;

用于存储所述处理器可执行指令的存储器;

其中,所述处理器被配置为执行所述指令,以实现如权利要求1至4中任一项所述的iOS终端的组件调用方法。

12.一种存储介质,其特征在于,当所述存储介质中的指令由电子设备的处理器执行时,使得所述电子设备能够执行如权利要求1至4中任一项所述的iOS终端的组件调用方法。

iOS组件的调用方法、装置、中间件、电子设备及介质

技术领域

[0001] 本发明涉及计算机技术领域，尤其涉及一种iOS组件的调用方法、装置、中间件、电子设备及存储介质。

背景技术

[0002] 随着移动设备操作系统(IOS，iPhone Operation System)项目开发的推进，新业务会不断的引入，导致移动终端应用程序(APP，Application)中的代码量越来越大，所有的代码集中在一个项目中，局部代码的变动往往需要修改与其关联的项目的代码，从而导致修改代码的时间越来越长，模块之间代码的耦合性越来越严重。

[0003] 相关技术中，基于统一资源定位符(URL，Uniform Resource Locator)运行时注册导致启动速度慢，参数传递比较单一，只能传递字符串，开发成本高，安全性不可控。中国移动集中运营平台Iop中间件与组件耦合严重，修改组件需要修改中间件，开发成本高，安全性也不可控。基于目标行为(Target-Action)操作，硬代码耦合更严重，修改组件需要修改中间件，开发成本高，安全性更不可控。

[0004] 因此，如何降低移动端组件间代码的耦合性及开发成本，提高开发效率，是目前有待解决的技术问题。

发明内容

[0005] 本发明提供一种iOS组件的业务调用方法、装置、中间件、电子设备及存储介质，以至少解决相关技术中由于移动端组件间代码的耦合性严重，导致开发成功高，开发效率低的技术的问题。本发明的技术方案如下：

[0006] 根据本发明实施例的第一方面，提供一种iOS终端的组件调用方法，包括：

[0007] 获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息；

[0008] 根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件；

[0009] 将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。

[0010] 可选的，所述根据所述第二组件的参数信息过调用函数调用对应的第二组件，包括：

[0011] 根据所述第二组件的参数信息读取对应文件lib包的头文件；

[0012] 获取所述头文件对应组件的含义以及应用程序接口API的调用方法；

[0013] 根据所述组件的含义以及API的调用方法通过调用函数调用对应的第二组件，并在所述组件调用中删除中间件的硬代码和分类文。

[0014] 可选的，在获取所述第一组件调用第二组件的调用请求之前，所述方法还包括：

[0015] 通过移动设备操作系统运行时iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式，其中，所述组件至少包括第一组件和第二组件；

[0016] 将所有组件中的每个组件的源码封装成对应的文件lib包；

[0017] 显示每个lib包的头文件，以便于根据所述头文件获取对应组件的含义以及应用程序接口API的调用方法。

[0018] 可选的，所述方法还包括：

[0019] 接收到外部调用应用程序APP；

[0020] 将所述调用APP的统一资源定位标识符URL映射成对应的类class进行调起。

[0021] 根据本发明实施例的第二方面，提供一种iOS终端的组件调用装置，包括：

[0022] 第一获取模块，用于获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息；

[0023] 调用模块，用于根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件；

[0024] 第一显示模块，用于将调用所述第二组件得到的结果通过所述第一组件显示给所述用户。

[0025] 可选的，所述调用模块，包括：

[0026] 读取模块，用于根据所述第二组件的参数信息读取对应文件lib包的头文件；

[0027] 第二获取模块，用于获取所述头文件对应组件的含义以及应用程序接口API的调用方法；

[0028] 调用子模块，用于根据所述组件的含义以及API的调用方法通过调用函数调用对应的第二组件，并在调用中删除中间件的硬代码和分类文。

[0029] 可选的，所述装置还包括：

[0030] 创建模块，用于在所述第一获取模块获取所述第一组件调用第二组件的调用请求之前，通过移动设备操作系统运行时iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式，其中，所述组件至少包括第一组件和第二组件；

[0031] 封装模块，用于将所有组件中的每个组件的源码封装成对应的文件lib包；

[0032] 第二显示模块，用于显示每个lib包的头文件，以便于根据所述头文件获取对应组件的含义以及应用程序接口API的调用方法。

[0033] 可选的，所述装置还包括：

[0034] 接收模块，用于接收到外部调用应用程序App；

[0035] 映射模块，用于将所述调用App的统一资源定位标识符URL映射成对应的类class进行调起。

[0036] 根据本发明实施例的第三方面，提供一种基于i OS终端的组件调用的中间件，包括：

[0037] 移动设备操作系统运行时iOS runtime模块，用于动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式；

[0038] 调用函数模块，用于获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息，根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件；将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。

[0039] 可选的，所述调用函数模块，还用于在根据所述第二组件的参数信息调用对应的第二组件中删除所述中间件的硬代码和分类文。

[0040] 根据本发明实施例的第四方面，本发明实施例还提供一种电子设备，包括：

[0041] 处理器；

[0042] 用于存储所述处理器可执行指令的存储器；

[0043] 其中，所述处理器被配置为执行所述指令，以实现如权利要求所述的iOS终端的组件调用方法。

[0044] 根据本发明实施例的第五方面，提供一种存储介质，当所述存储介质中的指令由电子设备的处理器执行时，使得所述电子设备能够执行如上所述的iOS终端的组件调用方法。

[0045] 根据本发明实施例的第六方面，提供根据一种计算机程序产品，当所述计算机程序产品中的指令由电子设备的处理器执行时，使得所述电子设备执行上述的iOS终端的组件调用方法。

[0046] 本发明的实施例提供的技术方案至少可以包括以下有益效果：

[0047] 本发明实施例中，在获取第一组件调用第二组件的调用请求时，根据所述调用请求中第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件；并将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。也就会说，本发明实施例中，通过中间件直接调用对应的组件进行信息传递，增加了各组件的独立性，降低了组件间的相互关联，消除组件间耦合以及组件和中间件的耦合性，降低了学习成本，且保证了安全行可控。

[0048] 应当理解的是，以上的一般描述和后文的细节描述仅是示例性和解释性的，并不能限制本发明。

附图说明

[0049] 此处的附图被并入说明书中并构成本说明书的一部分，示出了符合本发明的实施例，并与说明书一起用于解释本发明的原理，并不构成对本发明的不当限定。

[0050] 图1是根据一示例性实施例示出的一种iOS终端的组件调用方法的流程图。

[0051] 图2是根据一示例性实施例示出的一种iOS终端的组件调用方法的应用实例图。

[0052] 图3是根据一示例性实施例示出的一种iOS终端的组件调用装置的框图。

[0053] 图4是根据一示例性实施例示出的一种调用模块的框图。

[0054] 图5是根据一示例性实施例示出的一种iOS终端的组件调用装置的另一框图。

[0055] 图6是根据一示例性实施例示出的一种iOS终端的组件调用装置的又一框图。

[0056] 图7是根据一示例性实施例示出的一种基于iOS终端的组件调用的中间件的框图。

[0057] 图8是根据一示例性实施例示出的一种电子设备的结构框图。

[0058] 图9是根据一示例性实施例示出的一种用于iOS终端的组件调用的装置的框图。

具体实施方式

[0059] 为了使本领域普通人员更好地理解本发明的技术方案，下面将结合附图，对本发

明实施例中的技术方案进行清楚、完整地描述。

[0060] 需要说明的是，本发明的说明书和权利要求书及上述附图中的术语“第一”、“第二”等是用于区别类似的对象，而不必用于描述特定的顺序或先后次序。应该理解这样使用的数据在适当情况下可以互换，以便这里描述的本发明的实施例能够以除了在这里图示或描述的那些以外的顺序实施。以下示例性实施例中所描述的实施方式并不代表与本发明相一致的所有实施方式。相反，它们仅是与如所附权利要求书中所详述的、本发明的一些方面相一致的装置和方法的例子。

[0061] 图1是根据一示例性实施例示出的一种iOS终端的组件调用方法的流程图，如图1所示，该iOS终端的组件调用方法用于终端中，包括以下步骤：

[0062] 在步骤101中，获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息；

[0063] 在步骤102中，根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件；

[0064] 在步骤103中，将调用所述第二组件得到的数据通过所述第一组件展示给所述用户。

[0065] 本发明所述的iOS终端的组件调用方法可以应用于终端、服务器等，在此不作限制，其终端实施设备可以是智能手机，笔记本电脑、平板电脑等电子设备，在此不作限制。

[0066] 下面结合图1，对本发明实施例提供的一种iOS终端的组件调用方法的具体实施步骤进行详细说明。

[0067] 首先，执行步骤101，获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息。

[0068] 其中，该步骤中，用户点击浏览器或safari，或者H5，或APP等，后台可以检测到该用户的点击指令，比如用户打开车商通APP，查看库存中的车的图像，发起调用请求，比如该调用请求中可以包括车的名称，型号等参数。此时，中间件获取到用户通过第一组件调用第二组件的调用请求，即中间件获取用户点击车商通APP查看库存中车照片的调用请求。

[0069] 其中，所述调用请求可以包括：第二组件的参数信息，该第二组件的参数信息可以包括一个参数，也可以包括多个参数，本实施例不做限制。当然，还可以包括调用者(即用户)的参数信息，调用函数等，具体可以根据实际需要来定。

[0070] 其次，执行步骤102，根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件。

[0071] 该步骤中，先根据所述第二组件的参数信息读取对应文件lib包的头文件；再获取所述头文件对应组件的含义以及应用程序接口API的调用方法；最后，根据所述组件的含义以及API的调用方法通过调用函数NSInvocation调用对应的第二组件，并在所述组件调用中删除中间件的硬代码和分类文。中间件薄且轻，其作用专一且轻量移植性很高。

[0072] 其中，NSInvocation是调用函数的一种方式，它将调用者，函数名，参数封装到一个对象，然后通过一个invoke函数来执行被调用的函数，其思想就是命令者模式，将请求封装成对象。也就是说，NSInvocation是命令模式的一种实现，它包含选择器、方法签名、相应

的参数以及目标对象等;用于实现组件间的相互调用,并且在调用中删除中间件的硬代码和分类文件,比如中间件的种类或分类(category)文件。其中,所述方法签名,即方法所对应的返回值类型和参数类型。当NSInvocation被调用,它会在运行时通过目标对象去寻找对应的方法,从而确保唯一性,可以用[receiver message]来解释。实际开发过程中直接创建NSInvocation的情况不多见,这些事情通常交给系统来做。比如bang的JSPatch中arm64方法替换的实现就是利用runtime消息转发最后一步中的NSInvocation实现的。

[0073] 也就是说,本实施例中的调用函数,比如NSInvocation是一个消息调用类的函数,它包含了所有OC消息的成分:目标(target)、selector、参数以及返回值。NSInvocation可以将消息转换成一个对象,消息的每一个参数能够直接设定,而且当一个NSInvocation对象调度时返回值是可以自己设定的。一个NSInvocation对象能够重复的调度不同的目标(target),而且它的selector也能够设置为另外一个方法签名。NSInvocation遵守NSCoding协议,但是仅支持NSPortCoder编码,不支持归档型操作。需要说明的是,在面向对象中,方法也称消息。其中,NSInvocation的使用步骤为:根据方法(或消息中的参数信息)创建签名对象(NSMethodSignature对象),根据签名对象创建调用对象(NSInvocation对象),设置调用对象(NSInvocation对象)的相关信息,然后采用调用方法,获取方法返回值。

[0074] 其中,当调用组件的参数发生变化时,比如,第一组件第一次发起的调用第二组件的调用请求,该调用请求中可以只包括车的名称,比如,奥迪等;如果查找结果不理想,可以增加参数,再次发起调用请求,并在该调用请求中增加调用组件的参数,比如包括调用组件的名称和型号等,比如奥迪,Q5等。此时,中间件再次检测到调用请求时,直接按照该调用请求中的多个参加进行调用对应的组件。而现有技术中,在第一组件发起的调用请求中的参数发生变化时,中间件依据调用请求中的参数需要先通过category修改对应的参数和类,然后,才能向该参数对应的组件发起调用。由此可知,本发明与现有技术相比,特别是在调用组件的参数发生变化时,中间件不用修改本地对应的参数,直接根据调用请求中调用组件的参数调用对应的组件即可。

[0075] 最后,执行步骤103,将调用所述第二组件得到的数据通过所述第一组件展示给所述用户。

[0076] 中间件将调用所述第二组件得到的数据通过所述第一组件展示给所述用户。

[0077] 本示例性实施例示出的iOS终端的组件调用方法,获取第一组件调用第二组件的调用请求时,根据所述调用请求中第二组件的参数信息通过调用函数调用对应的第二组件;并将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。也就会说,本发明实施例中,通过中间件直接调用对应的组件进行信息传递,增加了各组件的独立性,降低了组件间的相互关联,消除组件间耦合以及组件和中间件的耦合性,降低了学习成本,且保证了安全行可控。

[0078] 可选的,在另一实施例中,该实施例在上述实施例的基础上,在获取所述第一组件调用第二组件的调用请求之前,所述方法还可以包括:

[0079] 1)通过移动设备操作系统运行时iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式,其中,所述组件至少包括第一组件和第二组件;

[0080] 该步骤中,Runtime又称为运行时,是一套用底层的C语言编写的API,Runtime为iOS内部的核心之一,平时编写的OC代码,底层都是基于它来实现的。

[0081] 也就是说,RunTime是运行时,OC就是运行时机制,其中最主要的是消息机制。对于C语言,函数的调用在编译的时候会决定调用哪个函数。对于OC的函数,属于动态调用过程,在编译的时候并不能决定真正调用哪个函数,只有在真正运行的时候才会根据函数的名称找到对应的函数来调用。本实施例的Runtime基本上是用C和汇编写的,为了动态系统的高效而作出的努力。

[0082] 其中,利用iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式,创建一个Person类等。其具体的创建方式,对于本领域技术人员已是熟知技术,在此不再赘述。

[0083] 2)将所有组件中的每个组件的源码封装成对应的文件lib包;

[0084] 该步骤中,将每个的组件的源码封装(即打包)成lib包,是为了节省内存空间,因为,源码比lib包占用的内存多。

[0085] 3)显示每个lib包的头文件,以便于根据所述头文件获取对应组件的含义以及应用程序接口API的调用方法。

[0086] 该步骤中,显示lib包的头文件的目的,就是在检测到组件调用请求中,方便根据调用请求中的参数读取lib包暴露在外的头文件,从而获取与该头文件对应的组件的含义和API的调用方法。

[0087] 可选的,在另一实施例中,该实施例在上述实施例的基础上,所述方法还可以包括:接收到外部调用应用程序APP;将所述调用APP的统一资源定位标识符URL映射成对应的类(class)进行调起。

[0088] 该实施例中,在中间件接收到外部调用APP时,将所述调用APP的统一资源定位标识符URL映射成对应的类class进行调起。

[0089] 其中,调起,就是一个APP调起另外一个APP并传值。也就是说,为了增加用户体验,可能要求在一个APP中打开另外一个APP的需求,一般分为三种:显式调用跳转,隐式调用跳转,URL Scheme跳转等。

[0090] 本发明实施例中,为了增加用户体验,中间件在接收到外部调用应用程序APP时,将所述调用APP的统一资源定位标识符URL映射成对应的类class进行调起,通过中间件直接调用对应的组件进行信息传递,增加了各组件的独立性,降低了组件间的相互关联,消除组件间耦合以及组件和中间件的耦合性,降低了学习成本,且保证了安全行可控。

[0091] 还请参阅图2,为本发明实施例提供的一种iOS终端的组件调用方法的应用实例图,如图所示,以h5,其他APP或safari中的中间件,iOS终端的模块A和模块B为例,中间件分别与模块A和模块B连接。其中,该中间件以包括:调用函数(以NSInvocation为例),运行时(runtime)为例,模块A中以包括组件A_Action,组件Page_1,组件Page_2,……为例,模块B中以包括组件B_Action,组件Page_1,组件Page_2,……为例,但在实际应用中,并不限于此。需要说明的是,图中两处打叉的双箭头,表示现有技术中,各组件之间相互连接,而本发明实施例中的模块A中的组件和模块B中的组件分别与中间件连接,具有单一的连接关系,所以,而图中打叉的category是现有技术中的分类、种类等,而本实施例中,在组件的相互调用时,删除中间件中的category的硬代码和分类文件,即不需要category,所以,删除了图中category,即打叉的category。

[0092] 如图所示,比如,中间件获取模块A中的组件A_Action调用模块B中的组件B_

Action的调用请求,所述调用请求包括:组件B的参数信息,该参数信息可以是一个,也可以是多个,NSInvocation根据所述组件B的参数信息直接调用对应的组件B,;将调用所述组件B得到的结果通过所述组件A进行显示。特别是当调用组件B的参数信息发生变化时(如,原来一个参数,后来增加为两个参数),此时,NSInvocation不用进行参数修改,直接根据该组件B的两个参数调用对应的组件。

[0093] 需要说明的是,对于方法实施例,为了简单描述,故将其都表述为一系列的动作组合,但是本领域技术人员应该知悉,本实施公开并不受所描述的动作顺序的限制,因为依据本发明,某些步骤可以采用其他顺序或者同时进行。其次,本领域技术人员也应该知悉,说明书中所描述的实施例均属于优选实施例,所涉及的动作并不一定是本发明所必须的。

[0094] 可选的,本发明实施例还提供一种iOS终端的组件调用装置,其结构框图如图3所示,所述装置包括:第一获取模块301,调用模块302和第一显示模块303,其中,

[0095] 该第一获取模块301,用于获取第一组件调用第二组件的调用请求,所述调用请求包括:第二组件的参数信息;

[0096] 调用模块该302,用于根据所述第二组件的参数信息通过调用函数调用对应的第二组件,其中,所述调用函数是一种消息调用类的函数,根据所述调用请求中的参数信息直接调用对应的组件;

[0097] 该第一显示模块303,用于将调用所述第二组件得到的结果通过所述第一组件显示给所述用户。

[0098] 可选的,在另一实施例中,该实施例在上述实施例的基础上,所述调用模块302包括:读取模块401,第二获取模块402和调用子模块403,其结构框图如图4所示,其中,

[0099] 该读取模块401,用于根据所述第二组件的参数信息读取对应文件lib包的头文件;

[0100] 该第二获取模块402,用于获取所述头文件对应组件的含义以及应用程序接口API的调用方法;

[0101] 该调用子模块403,用于根据所述组件的含义以及API的调用方法通过调用函数调用对应的第二组件,并在调用中删除中间件的硬代码和分类文。

[0102] 可选的,在另一实施例中,该实施例在上述实施例的基础上,所述装置还包括:创建模块501,封装模块502和第二显示模块503,其结构框图如图5所示,其中,

[0103] 该创建模块501,用于在所述第一获取模块获取所述第一组件调用第二组件的调用请求之前,通过移动设备操作系统运行时iOS runtime动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式,其中,所述组件至少包括第一组件和第二组件;

[0104] 该封装模块502,用于将所有组件中的每个组件的源码封装成对应的文件lib包;

[0105] 该第二显示模块503,用于显示每个lib包的头文件,以便于根据所述头文件获取对应组件的含义以及应用程序接口API的调用方法。

[0106] 可选的,在另一实施例中,该实施例在上述实施例的基础上,所述装置还可以包括:接收模块601和映射模块602,其中,

[0107] 该接收模块601,用于接收到外部调用应用程序App;

[0108] 该映射模块602,用于将所述调用App的统一资源定位标识符URL映射成对应的类class进行调起。

[0109] 可选的，本发明实施例还提供一种基于iOS终端的组件调用的中间件，其结构示意图如图7所示，其中，所述中间件包括：移动设备操作系统运行时(iOS runtime)模块701和调用函数(NSInvocation)模块702，其中，

[0110] 该移动设备操作系统运行时iOS runtime模块701，用于动态创建iOS终端的所有组件的类和动态创建所有组件的调用方式；

[0111] 该调用函数模块702，用于获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息，根据所述第二组件的参数信息调用对应的第二组件；将调用所述第二组件得到的结果通过所述第一组件展示给所述用户。

[0112] 可选的，在另一实施例中，该实施例在上述实施例的基础上，所述NSInvocation模块，还用于在根据所述第二组件的参数信息调用对应的第二组件中删除所述中间件的硬代码和分类文。

[0113] 关于上述实施例中的装置，其中各个模块执行操作的具体方式已经在有关该方法的实施例中进行了详细描述，相关之处参见方法实施例的部分说明即可，此处将不做详细阐述说明。

[0114] 可选的，本发明实施例还提供一种电子设备，包括：

[0115] 处理器；

[0116] 用于存储所述处理器可执行指令的存储器；

[0117] 其中，所述处理器被配置为执行所述指令，以实现如上所述的iOS终端的组件调用方法。

[0118] 可选的，本发明实施例还提供一种存储介质，当所述存储介质中的指令由电子设备的处理器执行时，使得所述电子设备能够执行如上所述的iOS终端的组件调用方法。

[0119] 在示例性实施例中，还提供了一种包括指令的存储介质，例如包括指令的存储器，上述指令可由装置的处理器执行以完成上述方法。可选地，存储介质可以是非临时性计算机可读存储介质，例如，所述非临时性计算机可读存储介质可以是ROM、随机存取存储器(RAM)、CD-ROM、磁带、软盘和光数据存储设备等。

[0120] 图8是根据一示例性实施例示出的一种电子设备800的框图。例如，电子设备800可以为移动终端也可以为服务器，本发明实施例中以电子设备为移动终端为例进行说明。例如，电子设备800可以是移动电话，计算机，数字广播终端，消息收发设备，游戏控制台，平板设备，医疗设备，健身设备，个人数字助理等。

[0121] 参照图8，电子设备800可以包括以下一个或多个组件：处理组件802，存储器804，电力组件806，多媒体组件808，音频组件810，输入/输出(I/O)的接口812，传感器组件814，以及通信组件816。

[0122] 处理组件802通常控制电子设备800的整体操作，诸如与显示，电话呼叫，数据通信，相机操作和记录操作相关联的操作。处理组件802可以包括一个或多个处理器820来执行指令，以完成上述的方法的全部或部分步骤。此外，处理组件802可以包括一个或多个模块，便于处理组件802和其他组件之间的交互。例如，处理组件802可以包括多媒体模块，以方便多媒体组件808和处理组件802之间的交互。

[0123] 存储器804被配置为存储各种类型的数据以支持在设备800的操作。这些数据的示例包括用于在电子设备800上操作的任何应用程序或方法的指令，联系人数据，电话簿数

据，消息，图片，视频等。存储器804可以由任何类型的易失性或非易失性存储设备或者它们的组合实现，如静态随机存取存储器(SRAM)，电可擦除可编程只读存储器(EEPROM)，可擦除可编程只读存储器(EPROM)，可编程只读存储器(PROM)，只读存储器(ROM)，磁存储器，快闪存储器，磁盘或光盘。

[0124] 电源组件806为电子设备800的各种组件提供电力。电源组件806可以包括电源管理系统，一个或多个电源，及其他与为电子设备800生成、管理和分配电力相关联的组件。

[0125] 多媒体组件808包括在所述电子设备800和用户之间的提供一个输出接口的屏幕。在一些实施例中，屏幕可以包括液晶显示器(LCD)和触摸面板(TP)。如果屏幕包括触摸面板，屏幕可以被实现为触摸屏，以接收来自用户的输入信号。触摸面板包括一个或多个触摸传感器以感测触摸、滑动和触摸面板上的手势。所述触摸传感器可以不仅感测触摸或滑动动作的边界，而且还检测与所述触摸或滑动操作相关的持续时间和压力。在一些实施例中，多媒体组件808包括一个前置摄像头和/或后置摄像头。当设备800处于操作模式，如拍摄模式或视频模式时，前置摄像头和/或后置摄像头可以接收外部的多媒体数据。每个前置摄像头和后置摄像头可以是一个固定的光学透镜系统或具有焦距和光学变焦能力。

[0126] 音频组件810被配置为输出和/或输入音频信号。例如，音频组件810包括一个麦克风(MIC)，当电子设备800处于操作模式，如呼叫模式、记录模式和语音识别模式时，麦克风被配置为接收外部音频信号。所接收的音频信号可以被进一步存储在存储器804或经由通信组件816发送。在一些实施例中，音频组件810还包括一个扬声器，用于输出音频信号。

[0127] I/O接口812为处理组件802和外围接口模块之间提供接口，上述外围接口模块可以是键盘，点击轮，按钮等。这些按钮可包括但不限于：主页按钮、音量按钮、启动按钮和锁定按钮。

[0128] 传感器组件814包括一个或多个传感器，用于为电子设备800提供各个方面的状态评估。例如，传感器组件814可以检测到设备800的打开/关闭状态，组件的相对定位，例如所述组件为电子设备800的显示器和小键盘，传感器组件814还可以检测电子设备800或电子设备800一个组件的位置改变，用户与电子设备800接触的存在或不存在，电子设备800方位或加速/减速和电子设备800的温度变化。传感器组件814可以包括接近传感器，被配置用来在没有任何的物理接触时检测附近物体的存在。传感器组件814还可以包括光传感器，如CMOS或CCD图像传感器，用于在成像应用中使用。在一些实施例中，该传感器组件814还可以包括加速度传感器，陀螺仪传感器，磁传感器，压力传感器或温度传感器。

[0129] 通信组件816被配置为便于电子设备800和其他设备之间有线或无线方式的通信。电子设备800可以接入基于通信标准的无线网络，如WiFi，运营商网络(如2G、3G、4G或5G)，或它们的组合。在一个示例性实施例中，通信组件816经由广播信道接收来自外部广播管理系统的广播信号或广播相关信息。在一个示例性实施例中，所述通信组件816还包括近场通信(NFC)模块，以促进短程通信。例如，在NFC模块可基于射频识别(RFID)技术，红外数据协会(IrDA)技术，超宽带(UWB)技术，蓝牙(BT)技术和其他技术来实现。

[0130] 在示例性实施例中，电子设备800可以被一个或多个应用专用集成电路(ASIC)、数字信号处理器(DSP)、数字信号处理设备(DSPD)、可编程逻辑器件(PLD)、现场可编程门阵列(FPGA)、控制器、微控制器、微处理器或其他电子元件实现，用于执行上述所示的iOS终端的组件调用方法。

[0131] 在示例性实施例中，还提供了一种包括指令的非临时性计算机可读存储介质，例如包括指令的存储器804，上述指令可由所述电子设备的处理器820执行以完成上述所示的iOS终端的组件调用方法。例如，所述非临时性计算机可读存储介质可以是ROM、随机存取存储器(RAM)、CD-ROM、磁带、软盘和光数据存储设备等。

[0132] 在示例性实施例中，还提供了一种计算机程序产品，当计算机程序产品中的指令由电子设备800的处理器820执行时，使得电子设备800执行上述所示的iOS终端的组件调用方法。

[0133] 图9是根据一示例性实施例示出的一种用于iOS终端的组件调用的装置900的框图。例如，装置900可以被提供为一服务器。参照图9，装置900包括处理组件922，其进一步包括一个或多个处理器，以及由存储器932所代表的存储器资源，用于存储可由处理组件922的执行的指令，例如应用程序。存储器932中存储的应用程序可以包括一个或一个以上的每一个对应于一组指令的模块。此外，处理组件922被配置为执行指令，以执行上述方法iOS终端的组件调用的方法。

[0134] 装置900还可以包括一个电源组件926被配置为执行装置900的电源管理，一个有线或无线网络接口950被配置为将装置900连接到网络，和一个输入输出(I/O)接口958。装置900可以操作基于存储在存储器932的操作系统，例如Windows ServerTM，Mac OS XTM，UnixTM，LinuxTM，FreeBSDTM或类似。

[0135] 本领域技术人员在考虑说明书及实践这里公开的发明后，将容易想到本发明的其它实施方案。本申请旨在涵盖本发明的任何变型、用途或者适应性变化，这些变型、用途或者适应性变化遵循本发明的一般性原理并包括本发明未公开的本技术领域中的公知常识或惯用技术手段。说明书和实施例仅被视为示例性的，本发明的真正范围和精神由下面的权利要求指出。

[0136] 应当理解的是，本发明并不局限于上面已经描述并在附图中示出的精确结构，并且可以在不脱离其范围进行各种修改和改变。本发明的范围仅由所附的权利要求来限制。

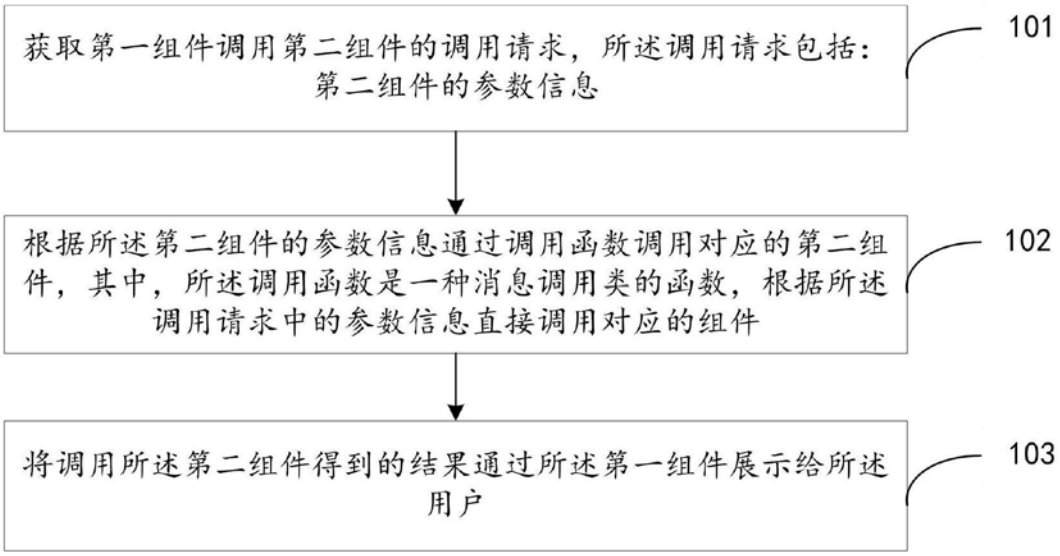

获取第一组件调用第二组件的调用请求，所述调用请求包括：第二组件的参数信息
101
根据所述第二组件的参数信息通过调用函数调用对应的第二组件，其中，所述调用函数是一种消息调用类的函数，根据所述调用请求中的参数信息直接调用对应的组件
102
将调用所述第二组件得到的结果通过所述第一组件展示给所述用户
103

图1

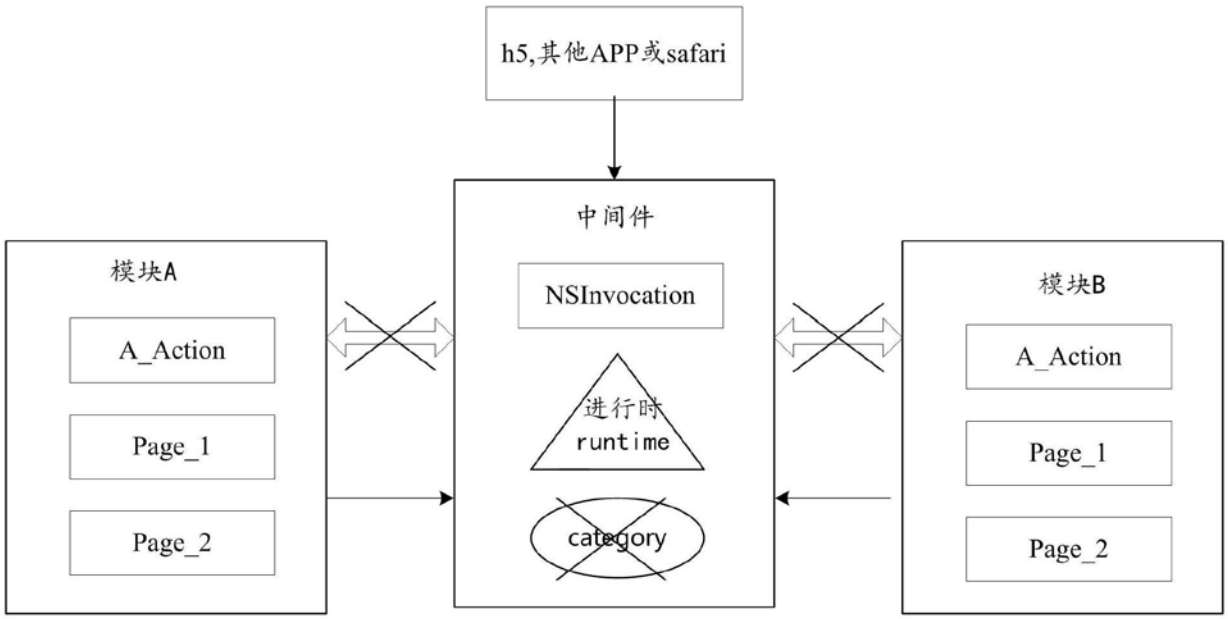

h5,其他APP或safari
中间件
NSInvocation
进行时
runtime
category
模块A
A_Action
Page_1
Page_2
模块B
A_Action
Page_1
Page_2

图2

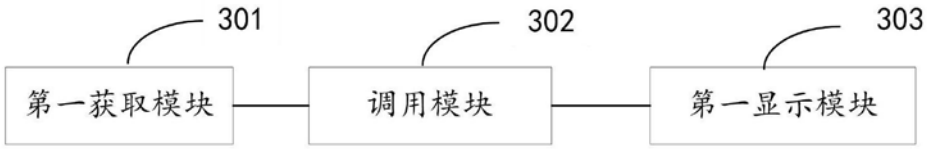

301
302
303
第一获取模块
调用模块
第一显示模块

图3

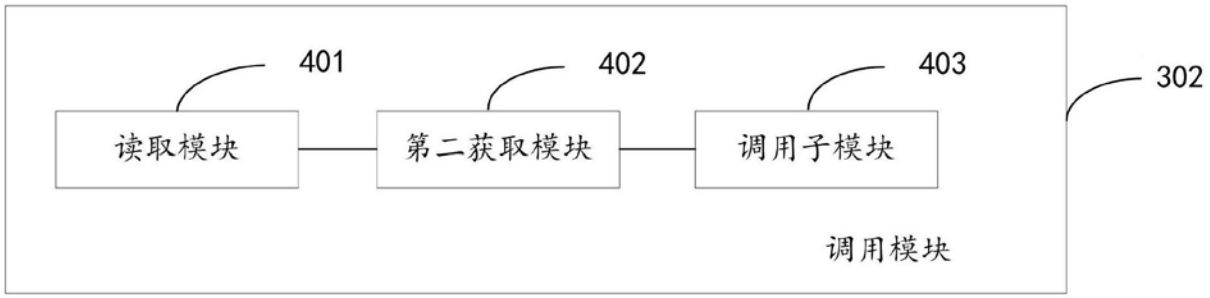

401
402
403
302
读取模块
第二获取模块
调用子模块
调用模块

图4

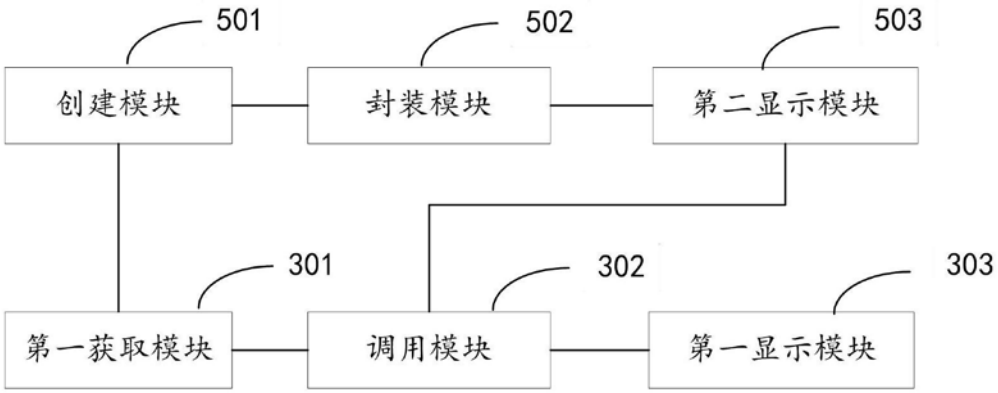

501
502
503
创建模块
封装模块
第二显示模块
301
302
303
第一获取模块
调用模块
第一显示模块

图5

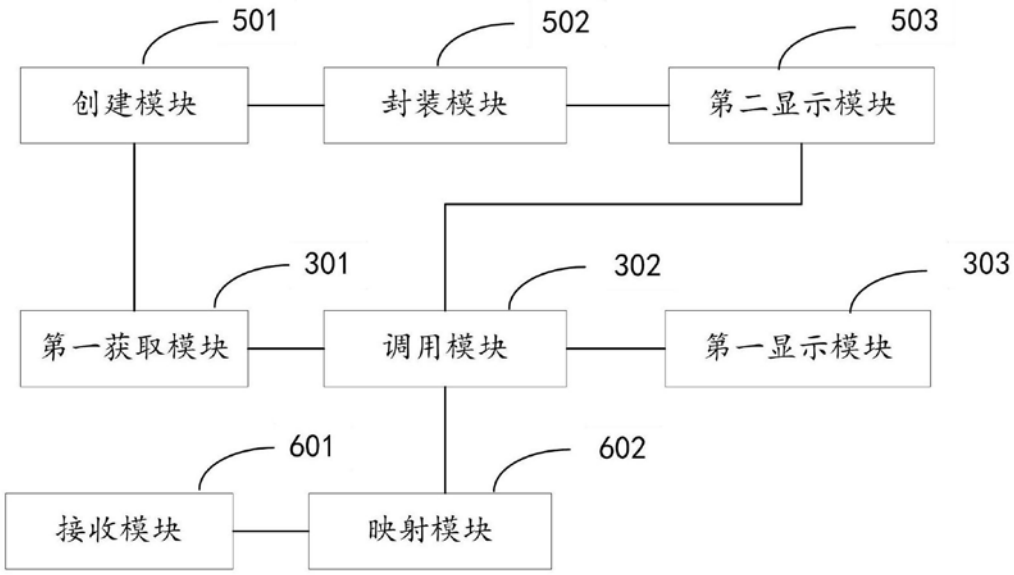

501
502
503
创建模块
封装模块
第二显示模块
301
302
303
第一获取模块
调用模块
第一显示模块
601
602
接收模块
映射模块

图6

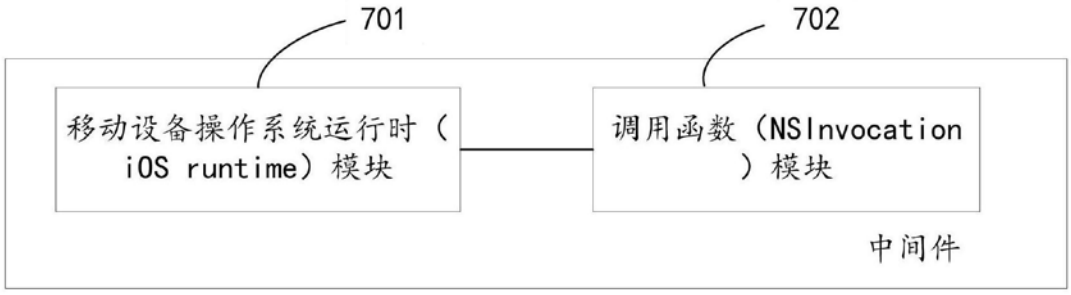

701
702
移动设备操作系统运行时（iOS runtime）模块
调用函数（NSInvocation）模块
中间件

图7

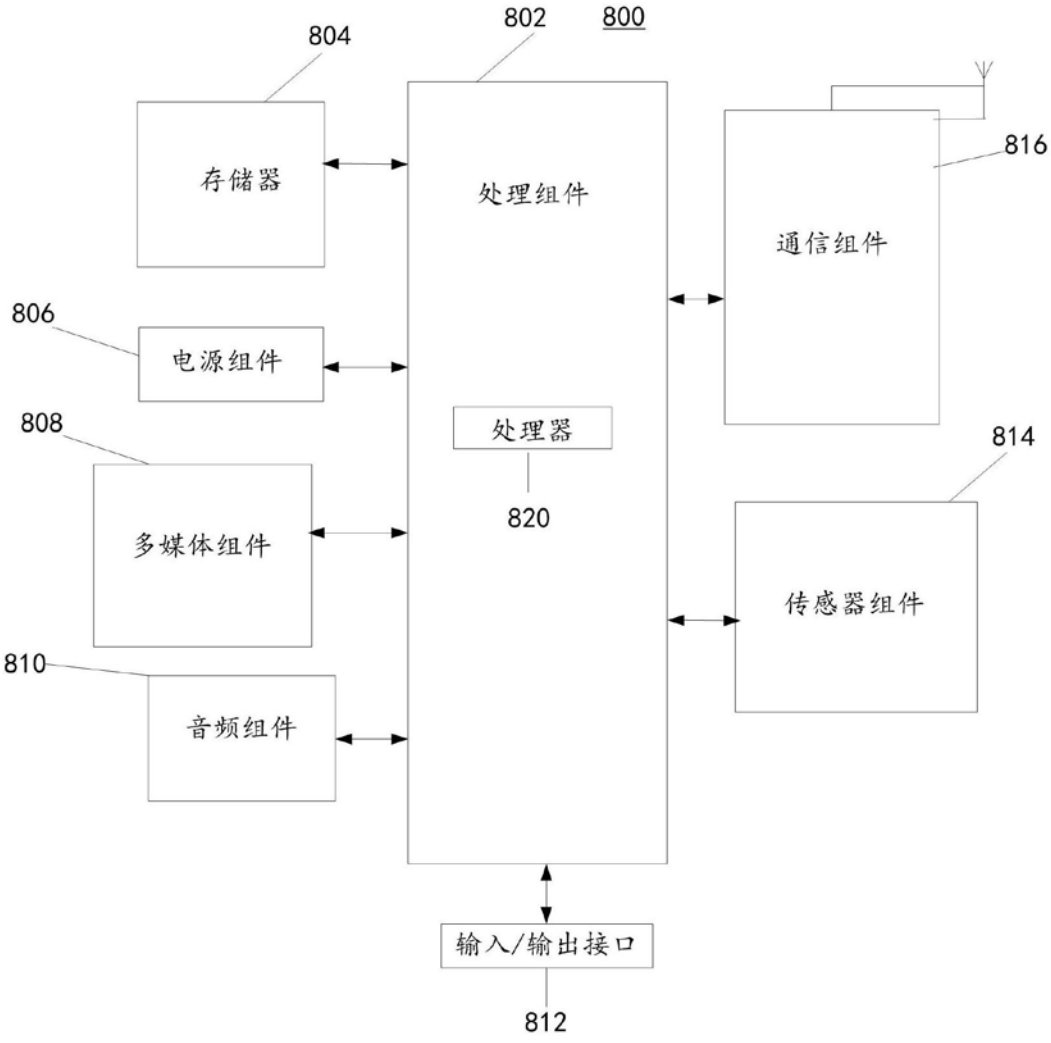

804
802
800
存储器
处理组件
816
通信组件
806
电源组件
处理器
820
808
814
多媒体组件
传感器组件
810
音频组件
输入/输出接口
812

图8

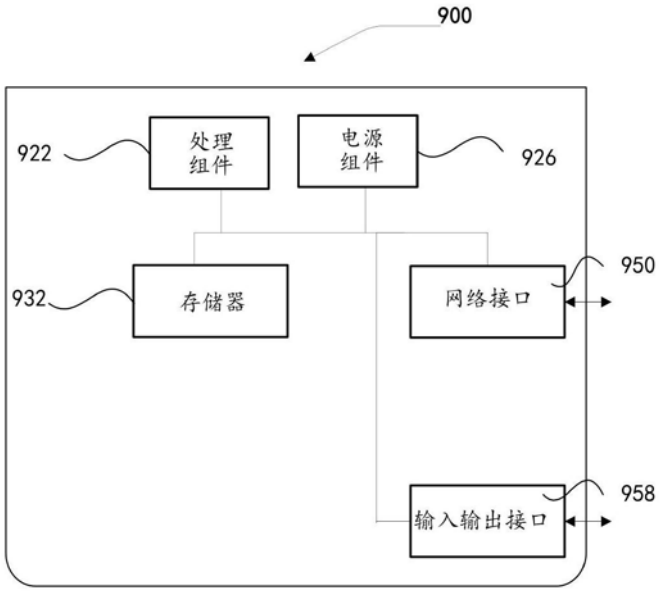


图9