(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 113434134 B
(45)授权公告日 2024.04.16

(21)申请号 202110688308.X
(22)申请日 2021.06.21
(65)同一申请的已公布的文献号
申请公布号 CN 113434134 A
(43)申请公布日 2021.09.24
(73)专利权人 北京达佳互联信息技术有限公司
地址 100085 北京市海淀区上地西路6号1幢1层101D1-7
(72)发明人 王少星
(74)专利代理机构 北京润泽恒知识产权代理有限公司 11319
专利代理师 李娜
(51)Int.Cl.
G06F 8/36(2018.01)
G06F 8/41(2018.01)

(56)对比文件
CN 110619220 A,2019.12.27
CN 112148926 A,2020.12.29
CN 112199086 A,2021.01.08
CN 112286529 A,2021.01.29
CN 112926008 A,2021.06.08

审查员 张诗浩

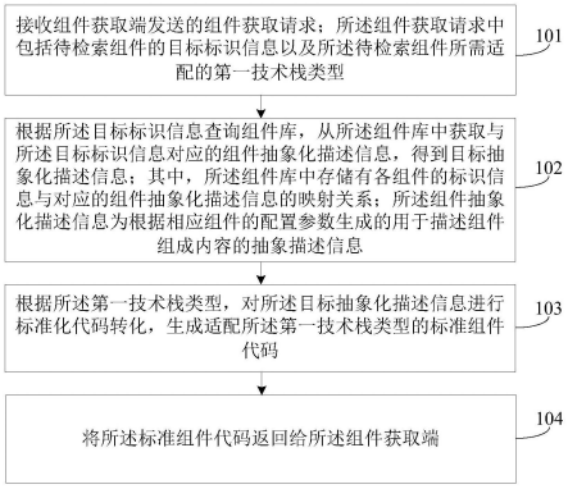
权利要求书3页 说明书15页 附图6页

(54)发明名称

组件处理方法、装置、电子设备及存储介质

(57)摘要

本公开提供了一种组件处理方法、装置、电子设备及存储介质，属于网络技术领域。本公开实施例中，可以接收组件获取端发送组件获取请求，组件获取请求包括待检索组件的目标标识信息以及待检索组件所需适配的第一技术栈类型。根据目标标识信息查询组件库，从组件库中获取与目标标识信息对应的组件抽象化描述信息，得到目标抽象化描述信息。根据第一技术栈类型，对目标抽象化描述信息进行标准化代码转化，生成适配第一技术栈类型的标准组件代码，将标准组件代码返回给组件获取端。这样，可以避免由于技术栈类型不匹配，导致需要二次改造，进而导致成本较大，组件复用率较低的问题。



CN 113434134 B

1.一种组件处理方法,其特征在于,应用于组件服务端,所述方法包括:

接收组件获取端发送的组件获取请求;所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型;

根据所述目标标识信息查询组件库,从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息,得到目标抽象化描述信息;其中,所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系;所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息;

根据所述第一技术栈类型,对所述目标抽象化描述信息进行标准化代码转化,生成适配所述第一技术栈类型的标准组件代码;

将所述标准组件代码返回给所述组件获取端;

其中,所述方法还包括:

根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息,包括:

对于任一所述组件,检测所述组件是否存在组件依赖,以及,根据所述组件的配置参数获取所述组件对应的结构化描述信息;

若所述组件存在组件依赖,则获取所述组件所依赖的组件;将所述结构化描述信息以及所述所依赖的组件,确定为所述组件抽象化描述信息;

若所述组件不存在组件依赖,则直接将所述结构化描述信息确定为所述组件抽象化描述信息。

2.根据权利要求1所述的方法,其特征在于,所述方法还包括:

获取各所述组件的标识信息;

将各所述组件的标识信息与所述组件抽象化描述信息对应存储至所述组件库,以生成所述映射关系。

3.根据权利要求2所述的方法,其特征在于,所述根据所述目标标识信息查询组件库,从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息,包括:

从所述组件库中各所述组件的标识信息中,查找与所述目标标识信息相匹配的标识信息;

将所述相匹配的标识信息对应的组件抽象化描述信息,确定为所述目标标识信息对应的组件抽象化描述信息。

4.根据权利要求1所述的方法,其特征在于,所述根据所述组件的配置参数获取所述组件对应的结构化描述信息,包括:

检测所述组件的组件类型是否为数据驱动类型;

在所述组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,根据所述组件的配置参数将所述组件转换为所述结构化描述信息;所述第二技术栈类型为所述组件当前适配的技术栈类型。

5.根据权利要求4所述的方法,其特征在于,所述在所述组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,根据所述组件的配置参数将所述组件转换为所述结构化描述信息之前,所述方法还包括:

在所述组件类型不为数据驱动类型的情况下,对于所述组件进行组件类型改造,以将所述组件的组件类型修改为所述数据驱动类型。

6.根据权利要求1所述的方法,其特征在于,所述根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息,还包括:接收用户端发送的根据各所述组件的配置参数生成的组件抽象化描述信息;

所述获取各所述组件的标识信息,包括:接收所述用户端发送的各所述组件的标识信息。

7.根据权利要求2至6任一所述的方法,其特征在于,所述标识信息包括组件名称以及所述组件的参数信息。

8.一种组件处理装置,其特征在于,应用于组件服务端,所述装置包括:

接收模块,被配置为接收组件获取端发送的组件获取请求;所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型;

第一获取模块,被配置为根据所述目标标识信息查询组件库,从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息,得到目标抽象化描述信息;其中,所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系;所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息;

转化模块,被配置为根据所述第一技术栈类型,对所述目标抽象化描述信息进行标准化代码转化,生成适配所述第一技术栈类型的标准组件代码;

返回模块,被配置为将所述标准组件代码返回给所述组件获取端;

其中,所述所述装置还包括:

第二获取模块,被配置为根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息;

所述第二获取模块,具体被配置为:

对于任一所述组件,检测所述组件是否存在组件依赖,以及,根据所述组件的配置参数获取所述组件对应的结构化描述信息;

若所述组件存在组件依赖,则获取所述组件所依赖的组件;将所述结构化描述信息以及所述所依赖的组件,确定为所述组件抽象化描述信息;

若所述组件不存在组件依赖,则直接将所述结构化描述信息确定为所述组件抽象化描述信息。

9.根据权利要求8所述的装置,其特征在于,所述装置还包括:

第二获取模块,还被配置为获取各所述组件的标识信息;

存储模块,被配置为将各所述组件的标识信息与所述组件抽象化描述信息对应存储至所述组件库,以生成所述映射关系。

10.根据权利要求9所述的装置,其特征在于,所述第一获取模块,被具体配置为:

从所述组件库中各所述组件的标识信息中,查找与所述目标标识信息相匹配的标识信息;

将所述相匹配的标识信息对应的组件抽象化描述信息,确定为所述目标标识信息对应的组件抽象化描述信息。

11.根据权利要求8所述的装置,其特征在于,所述第二获取模块,还具体被配置为:

检测所述组件的组件类型是否为数据驱动类型;

在所述组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,根据

所述组件的配置参数将所述组件转换为所述结构化描述信息;所述第二技术栈类型为所述组件当前适配的技术栈类型。

12.根据权利要求11所述的装置,其特征在于,所述装置还包括:

改造模块,被配置为在所述组件类型不为数据驱动类型的情况下,对于所述组件进行组件类型改造,以将所述组件的组件类型修改为所述数据驱动类型。

13.根据权利要求10所述的装置,其特征在于,第二获取模块,具体被配置为:

接收用户端发送的根据各所述组件的配置参数生成的组件抽象化描述信息;

接收所述用户端发送的各所述组件的标识信息。

14.根据权利要求9至13任一所述的装置,其特征在于,所述组件标识信息包括组件名称以及所述组件的参数信息。

15.一种电子设备,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7中任一项所述的组件处理方法。

16.一种存储介质,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,使得所述电子设备执行如权利要求1至7中任一项所述的组件处理方法。

17.一种计算机程序产品,其特征在于,所述计算机程序产品包括可读性程序指令,所述可读性程序指令由电子设备的处理器执行时,使得所述电子设备执行如权利要求1至7中任一项所述的组件处理方法。

组件处理方法、装置、电子设备及存储介质

技术领域

[0001] 本公开属于网络技术领域，特别是涉及一种组件处理方法、装置、电子设备及存储介质。

背景技术

[0002] 目前，为了方便对代码进行处理，往往会对代码进行组件化。其中，一个组件可以是一段具有特定功能和样式的代码，该组件作为一个整体，无论放在哪里去使用，都具有一样的功能和样式，从而可以方便复用。为了提高组件的利用率，往往会将组件代码统一存储至组件库，以供使用。

[0003] 现有方式中，通常是直接将组件本身存储至组件库中，组件获取端可以根据需要从中获取所需的组件进行使用。这样，有时会出现获取到的组件所采用的技术栈类型与当前所需使用的技术栈类型不适配，进而使得用户需要进一步对获取到的组件进行二次开发，成本较大，组件的复用率较低。

发明内容

[0004] 为克服相关技术中存在的问题，本公开提供一种组件处理方法、装置、电子设备及存储介质。

[0005] 依据本公开的第一方面，提供了一种组件处理方法，应用于组件服务端，所该方法包括：

[0006] 接收组件获取端发送的组件获取请求；所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型；

[0007] 根据所述目标标识信息查询组件库，从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息，得到目标抽象化描述信息；其中，所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系；所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息；

[0008] 根据所述第一技术栈类型，对所述目标抽象化描述信息进行标准化代码转化，生成适配所述第一技术栈类型的标准组件代码；

[0009] 将所述标准组件代码返回给所述组件获取端。

[0010] 可选的，所述方法还包括：

[0011] 根据各所述组件的配置参数，获取各所述组件对应的组件抽象化描述信息，以及，获取各所述组件的标识信息；

[0012] 将各所述组件的标识信息与所述组件抽象化描述信息对应存储至所述组件库，以生成所述映射关系。

[0013] 可选的，所述根据所述目标标识信息查询组件库，从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息，包括：

[0014] 从所述组件库中各所述组件的标识信息中，查找与所述目标标识信息相匹配的标

识信息；

[0015] 将所述相匹配的标识信息对应的组件抽象化描述信息，确定为所述目标标识信息对应的组件抽象化描述信息。

[0016] 可选的，所述根据各所述组件的配置参数，获取各所述组件对应的组件抽象化描述信息，包括：

[0017] 对于任一所述组件，检测所述组件是否存在组件依赖，以及，根据所述组件的配置参数获取所述组件对应的结构化描述信息；

[0018] 若所述组件存在组件依赖，则获取所述组件所依赖的组件；将所述结构化描述信息以及所述所依赖的组件，确定为所述组件抽象化描述信息；

[0019] 若所述组件不存在组件依赖，则直接将所述结构化描述信息确定为所述组件抽象化描述信息。

[0020] 可选的，所述根据所述组件的配置参数获取所述组件对应的结构化描述信息，包括：

[0021] 检测所述组件的组件类型是否为数据驱动类型；

[0022] 在所述组件类型为数据驱动类型的情况下，以第二技术栈类型对应的转换工具，根据所述组件的配置参数将所述组件转换为所述结构化描述信息；所述第二技术栈类型为所述组件当前适配的技术栈类型。

[0023] 可选的，所述在所述组件类型为数据驱动类型的情况下，以第二技术栈类型对应的转换工具，根据所述组件的配置参数将所述组件转换为所述结构化描述信息之前，所述方法还包括：

[0024] 在所述组件类型不为数据驱动类型的情况下，对于所述组件进行组件类型改造，以将所述组件的组件类型修改为所述数据驱动类型。

[0025] 可选的，所述根据各所述组件的配置参数，获取各所述组件对应的组件抽象化描述信息，包括：接收用户端发送的根据各所述组件的配置参数生成的组件抽象化描述信息；

[0026] 所述获取各所述组件的标识信息，包括：接收所述用户端发送的各所述组件的标识信息。

[0027] 可选的，所述标识信息包括组件名称以及所述组件的参数信息。

[0028] 依据本公开的第二方面，提供了一种组件处理装置，应用于组件服务端，所述装置包括：

[0029] 接收模块，被配置为接收组件获取端发送的组件获取请求；所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型；

[0030] 第一获取模块，被配置为根据所述目标标识信息查询组件库，从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息，得到目标抽象化描述信息；其中，所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系；所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息；

[0031] 转化模块，被配置为根据所述第一技术栈类型，对所述目标抽象化描述信息进行标准化代码转化，生成适配所述第一技术栈类型的标准组件代码；

[0032] 返回模块，被配置为将所述标准组件代码返回给所述组件获取端。

[0033] 可选的，所述装置还包括：

[0034] 第二获取模块,被配置为根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息,以及,获取各所述组件的标识信息;

[0035] 存储模块,被配置为将各所述组件的标识信息与所述组件抽象化描述信息对应存储至所述组件库,以生成所述映射关系。

[0036] 可选的,所述第一获取模块,被具体配置为:

[0037] 从所述组件库中各所述组件的标识信息中,查找与所述目标标识信息相匹配的标识信息;

[0038] 将所述相匹配的标识信息对应的组件抽象化描述信息,确定为所述目标标识信息对应的组件抽象化描述信息。

[0039] 可选的,所述第二获取模块,具体被配置为:

[0040] 对于任一所述组件,检测所述组件是否存在组件依赖,以及,根据所述组件的配置参数获取所述组件对应的结构化描述信息;

[0041] 若所述组件存在组件依赖,则获取所述组件所依赖的组件;将所述结构化描述信息以及所述所依赖的组件,确定为所述组件抽象化描述信息;

[0042] 若所述组件不存在组件依赖,则直接将所述结构化描述信息确定为所述组件抽象化描述信息。

[0043] 可选的,所述第二获取模块,还具体被配置为:

[0044] 检测所述组件的组件类型是否为数据驱动类型;

[0045] 在所述组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,根据所述组件的配置参数将所述组件转换为所述结构化描述信息;所述第二技术栈类型为所述组件当前适配的技术栈类型。

[0046] 可选的,所述装置还包括:

[0047] 改造模块,被配置为在所述组件类型不为数据驱动类型的情况下,对于所述组件进行组件类型改造,以将所述组件的组件类型修改为所述数据驱动类型。

[0048] 可选的,第二获取模块,具体被配置为:

[0049] 接收用户端发送的根据各所述组件的配置参数生成的组件抽象化描述信息;

[0050] 接收所述用户端发送的各所述组件的标识信息。

[0051] 可选的,所述组件标识信息包括组件名称以及所述组件的参数信息。

[0052] 依据本公开的第三方面,提供了一种电子设备,包括:

[0053] 处理器;

[0054] 用于存储所述处理器可执行指令的存储器;

[0055] 其中,所述处理器被配置为执行所述指令,以实现如第一方面中任一项所述的组件处理方法。

[0056] 依据本公开的第四方面,提供了一种存储介质,当所述存储介质中的指令由电子设备的处理器执行时,使得所述电子设备执行如第一方面中任一项所述的组件处理方法。

[0057] 依据本公开的第五方面,提供了一种计算机程序产品,所述计算机程序产品包括可读性程序指令,所述可读性程序指令由电子设备的处理器执行时,使得所述电子设备执行如第一方面中任一项所述的组件处理方法。

[0058] 本公开相比于相关技术,具有如下的优点和积极效果:

[0059] 本公开实施例提供的组件处理方法，可以接收组件获取端发送的组件获取请求，组件获取请求中包括待检索组件的目标标识信息以及待检索组件所需适配的第一技术栈类型。然后，根据目标标识信息查询组件库，从组件库中获取与目标标识信息对应的组件抽象化描述信息，得到目标抽象化描述信息。其中，组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系，组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息。根据第一技术栈类型，对目标抽象化描述信息进行标准化代码转化，生成适配第一技术栈类型的标准组件代码，将标准组件代码返回给组件获取端。这样，通过向组件获取端返回适配第一技术栈类型的标准组件代码，可以避免由于技术栈类型不匹配，导致需要二次改造，进而导致成本较大，组件复用率较低的问题。

[0060] 上述说明仅是本公开技术方案的概述，为了能够更清楚了解本公开的技术手段，而可依照说明书的内容予以实施，并且为了让本公开的上述和其它目的、特征和优点能够更明显易懂，以下特举本公开的具体实施方式。

附图说明

[0061] 通过阅读下文优选实施方式的详细描述，各种其他的优点和益处对于本领域普通技术人员将变得清楚明了。附图仅用于示出优选实施方式的目的，而并不认为是对本公开的限制。而且在整个附图中，用相同的参考符号表示相同的部件。在附图中：

[0062] 图1是本公开实施例提供的一种组件处理方法的步骤流程图；

[0063] 图2是本公开实施例提供的一种存储过程示意图；

[0064] 图3是一种现有方式的组件获取过程示意图；

[0065] 图4是本公开实施例提供的一种组件获取过程示意图；

[0066] 图5是本公开实施例提供的一种组件处理装置的框图；

[0067] 图6是根据一示例性实施例示出的一种用于组件处理的装置的框图；

[0068] 图7是根据一示例性实施例示出的一种用于组件处理的装置的框图。

具体实施方式

[0069] 下面将参照附图更详细地描述本公开的示例性实施例。虽然附图中显示了本公开的示例性实施例，然而应当理解，可以以各种形式实现本公开而不应被这里阐述的实施例所限制。相反，提供这些实施例是为了能够更透彻地理解本公开，并且能够将本公开的范围完整的传达给本领域的技术人员。

[0070] 图1是本公开实施例提供的一种组件处理方法的步骤流程图，该方法可以应用于组件服务端，如图1所示，该方法可以包括：

[0071] 步骤101、接收组件获取端发送的组件获取请求；所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型。

[0072] 本公开实施例中，待获取组件可以是组件获取端需要使用的组件，目标标识信息可以是待检索组件的标识信息，目标标识信息可以用于向组件服务端表征组件获取端当前所需要的组件的相关信息。示例的，用户可以通过组件获取端登录组件服务端，以通过目标标识信息以及第一技术栈类型，进行搜索。进一步地，第一技术栈类型以及目标标识信息可以是组件获取端根据实际需求设置的。其中，组件获取端可以向用户显示一选择界面，然后

将用户在选择界面中选中的技术栈类型作为第一技术栈类型。进一步地,本公开实施例中的标识信息可以包括组件名称以及组件的参数信息。这样,一定程度上可以使得用于从组件服务端中检测组件的检索信息更丰富,进而一定程度上可以提高检索的便捷性。当然,标识信息还可以进一步包括其他信息,本公开实施例对此不作限定。其中,组件的参数信息可以为组件的输入参数。示例的,标识信息可以包括组件名称“登录”,也可以进一步包括“用户名+密码”这样的关键字段级别的输入参数。

[0073] 步骤102、根据所述目标标识信息查询组件库,从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息,得到目标抽象化描述信息;其中,所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系;所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息。

[0074] 本步骤中,组件库可以位于组件服务端内部,也可以独立设置于组件服务端外部,本公开实施例对此不作限定。示例的,组件服务端可以为预设的服务器,该服务器中可以配置有用于存储组件的组件抽象化描述信息的组件库。示例的,组件对应的组件抽象化描述信息可以包括根据配置参数生成的抽象语法树(Abstract Syntax Tree,AST)文件。AST文件可以描述组件组成内容,属于对抽象逻辑的具象化定义实体(Schema),类似java编译阶段的字节码文件。其中,Schema可以用于描述抽象数据的定义,AST文件本质上可以理解为一种Schema信息。AST文件可以是代码的语法结构的树状表示。抽象语法树并不依赖于源语言的语法,因此,抽象语法树并不会表示出真实语法出现的每一个细节,而是以树状的形式表现编程语言的语法结构,可以用于描述代码程序的逻辑,描述抽象数据的定义,抽象语法树上的每个节点都表示源代码中的一种结构。进一步地,组件的技术栈类型可以指的是编写组件时,实现组件的前端功能的不同方式。示例的,技术栈类型可以包括Vue、React和Angular,等等。不同技术栈类型的处理思想往往一致,但是具体的实现层面的执行方式不同。进一步地,由于组件抽象化描述信息可以用于描述组件组成内容的抽象描述信息,即,组件抽象化描述信息可以表征组件的语法结构。而不同技术栈类型中实现同一功能的处理思想往往一致。因此,实际应用场景中,可以将组件抽象化描述信息转换为适配各种不同技术栈类型的组件代码,即,组件抽象化描述信息可以用来转换为适配不同技术栈类型的组件代码。

[0075] 进一步地,可以将目标标识信息与组件库中存储的各个标识信息进行比对,以筛选出与目标标识信息对应的组件抽象化描述信息,进而得到目标抽象化描述信息。

[0076] 步骤103、根据所述第一技术栈类型,对所述目标抽象化描述信息进行标准化代码转化,生成适配所述第一技术栈类型的标准组件代码。

[0077] 本公开实施例中,对应不同技术栈类型,将目标抽象化描述信息代码化的方式可以存在差异。具体转换时,可以遵循第一技术栈类型预先规定的代码语法,对目标抽象化描述信息进行转换,进而确保转换后得到的标准组件代码可以正常适配第一技术栈类型。

[0078] 步骤104、将所述标准组件代码返回给所述组件获取端。

[0079] 本公开实施例中,由于标准组件代码是适配组件获取端所需的第一技术栈类型的组件代码,因此,通过将标准组件代码返回给组件获取端,使得组件获取端可以直接使用该标准组件代码,进而使得组件获取端无需再进行额外的二次开发。

[0080] 本公开实施例提供的组件处理方法,可以接收组件获取端发送的组件获取请求,

组件获取请求中包括待检索组件的目标标识信息以及待检索组件所需适配的第一技术栈类型。然后,根据目标标识信息查询组件库,从组件库中获取与目标标识信息对应的组件抽象化描述信息,得到目标抽象化描述信息。其中,组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系,组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息。根据第一技术栈类型,对目标抽象化描述信息进行标准化代码转化,生成适配第一技术栈类型的标准组件代码,将标准组件代码返回给组件获取端。这样,通过向组件获取端返回适配第一技术栈类型的标准组件代码,可以避免由于技术栈类型不匹配,导致需要二次改造,进而导致成本较大,组件复用率较低的问题。

[0081] 同时,由于技术栈类型不同,或者同一技术栈类型的组件写法(例如,vue技术栈类型包括class base写法和Object base写法)不同,又或者组件开发标准规范不统一,因此,可能会出现调用方式不统一的问题。本公开实施例中,通过向组件获取端返回以目标抽象化描述文件转换得到的标准组件代码,一定程度上可以避免组件获取端在后续使用组件时,出现调用方式不统一的问题。

[0082] 可选的,本公开实施例还可以包括下述步骤,以构建组件库中的映射关系:

[0083] 步骤S21、根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息,以及,获取各所述组件的标识信息。

[0084] 本公开实施例中,组件抽象化描述信息可以用于转换为适配不同技术栈类型的组件。这些组件可以为需要存储至组件服务端的组件代码,组件可以具体为代码包的形式。组件可以为前端组件,组件的数量可以为多个。

[0085] 进一步地,组件抽象化描述信息以及标识信息可以是用户端发送给组件服务端的。即,组件服务端通过接收用户端发送的根据各组件的配置参数生成的组件抽象化描述信息,以及,接收用户端发送的各组件的标识信息,即可实现获取。示例的,用户端可以根据各组件的配置参数生成组件抽象化描述信息,然后从组件的配置文件中读取标识信息,然后将组件抽象化描述信息以及标识信息一并发送给组件服务端。例如,可以通过组件上传工具,将组件抽象化描述信息以及标识信息上传至组件服务端。相应地,组件服务端可以对接收到上传信息进行类型分析,以从中确定出标识信息以及组件抽象化描述信息,进而方便后续存储。这样,组件服务端中不再保存组件本身的代码,而是保存组件的组件抽象化描述信息,用户端仅需发送组件抽象化描述信息,即可实现存储操作,进而一定程度上可以确保存储效率。且组件服务端仅需通过接收操作即可实现获取,进而一定程度上可以确保组件服务端的处理效率。当然,也可以是由组件服务端根据组件的配置参数自主获取组件抽象化描述信息,以及自主获取标识信息。又或者,组件服务端也可以是仅接收用户端发送的组件抽象化描述信息,并自主获取标识信息,本公开实施例对此不作限定。

[0086] 步骤S22、将各所述组件的标识信息与所述组件抽象化描述信息对应存储至所述组件库,以生成所述映射关系。

[0087] 本公开实施例中,标识信息可以是能够指代组件的信息,示例的,标识信息可以为组件的名称,编号,相关参数,输入属性,等等。进一步地,存储时,可以将各组件的标识信息与组件抽象化描述信息按照一一对应的方式,分别进行记录并保存,即,存储至组件库中。例如,可以将标识信息作为键(key),将组件抽象化描述信息作为值(value),以key-value的方式对应存储。通过一一对应存储,即可形成映射关系。

[0088] 本公开实施例中,根据各组件的配置参数,获取各组件对应的组件抽象化描述信息,以及,获取各组件的标识信息,将各组件的标识信息与组件抽象化描述信息对应存储至组件库,即可生成映射关系,进而一定程度上可以确保生成效率。同时,通过生成映射关系可以方便组件服务端从中查找目标抽象化描述信息。

[0089] 可选的,上述根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息的操作,可以具体包括:

[0090] 步骤S31、对于任一所述组件,检测所述组件是否存在组件依赖,以及,根据所述组件的配置参数获取所述组件对应的结构化描述信息。

[0091] 本步骤中,可以组件的配置参数可以是组件中的可配置参数,进一步地,可以基于组件的代码内容以及其中包含的可配置参数,将该组件转换为AST文件,进而得到该组件对应的结构化描述信息。进一步地,检测组件是否存在组件依赖时,可以基于预设工具(例如,组件上传工具),对组件进行递归的依赖分析。示例的,组件依赖可以包括在特定配置文件(例如,package.json文件)里面指定的依赖,以及,对组件所在工程的全局组件的依赖。其中,package.json文件可以设置在组件所属项目的根目录下,其中还可以定义该项目所需要的各种模块,以及项目的配置信息。全局组件的依赖可以是通过组件包的入口文件递归分析后得到的。

[0092] 步骤S32、若所述组件存在组件依赖,则获取所述组件所依赖的组件;将所述结构化描述信息以及所述所依赖的组件,确定为所述组件抽象化描述信息。

[0093] 示例的,如果组件依赖为在package.json文件里面指定的依赖,则可以以引用的方式,得到所依赖的组件。如果组件依赖为对全局组件的依赖,则可以从存放工程依赖的子包的指定文件夹(例如,上层的node_module文件夹)中提取所依赖的组件,并进行记录。需要说明的是,本公开实施例中,还可以进一步记录所依赖组件的版本信息。在无法获取到所依赖的组件时,可以向用户发送指定请求,以使用户指定所依赖的组件。

[0094] 进一步地,在组件服务端通过接收用户端发送的组件抽象化描述信息的的情况下,即,由用户端执行根据各组件的配置参数,获取各组件对应的组件抽象化描述信息的具体操作的情况下,用户端可以在本地设备上进行检测,获取等操作,以实现获取组件所依赖的组件的操作。在组件存储端自己根据各组件的配置参数,获取各组件对应的组件抽象化描述信息的具体操作情况下,组件服务端可以访问存储有特定配置文件、指定文件夹等所依赖的组件的信息的设备,实现获取所依赖的组件。或者,也可以提前将这些信息发送给组件服务端,以便于组件服务端在本地实现获取所依赖的组件的操作。

[0095] 进一步地,实际应用场景中,被依赖的组件往往为通用组件,即,可以适配各种技术栈类型的组件。因此,本公开实施例中无需对所依赖的组件进行转换,可以直接将结构化描述信息以及所依赖的组件作为组件抽象化描述信息。

[0096] 步骤S33、若所述组件不存在组件依赖,则直接将所述结构化描述信息确定为所述组件抽象化描述信息。

[0097] 本步骤中,如果组件不存在组件依赖,则可以认为仅提供结构化描述信息,可以确保后续用户能够正常使用该组件。因此,可以直接将结构化描述信息确定为组件抽象化描述信息。这样,无需进一步执行额外操作,一定程度上可以确保获取组件抽象化描述信息的效率。

[0098] 本公开实施例中,通过获取组件对应的结构化描述信息,检测组件是否存在组件依赖,在组件存在组件依赖的情况下,获取组件所依赖的组件。最后,将结构化描述信息以及所依赖的组件,确定为组件抽象化描述信息。这样,可以确保生成的组件抽象化描述信息的完整性,避免缺少组件的依赖信息,导致其他用户无法正常使用组件服务端中存储的组件,进而导致组件利用率降低的问题。

[0099] 可选的,上述步骤S21中根据所述组件的配置参数获取所述组件对应的结构化描述信息的操作,可以具体包括:

[0100] 步骤S41、检测所述组件的组件类型是否为数据驱动类型。

[0101] 本步骤中,可以先获取组件。示例的,用户端可以将上传组件路径指定为组件所在目录,以将该组件发送给组件服务端。示例的,假设用户端为用户使用的电脑,在电脑本地中存储有组件A,那么用户可以控制用户端,利用组件上传工具将上传组件路径指定为组件A在该电脑中的所在目录。其中,该组件上传工具可以是组件服务端提供的。相应地,组件服务端可以通过接收用户端发送的组件,实现获取组件。在组件服务端通过接收组件存储端为用户端发送的组件抽象化描述信息的的情况下,即,由用户端执行根据各组件的配置参数,获取各组件对应的组件抽象化描述信息的具体操作的情况下,用户端可以直接读取本地的组件,实现获取组件。

[0102] 进一步地,在传统的组件的数据交互方式中,是获取到数据之后,操作DOM来改变视图。前端交互要改变数据时,又要重复一次获取数据-操作DOM的步骤,而操作DOM是一个繁琐的过程且易出错。其中,DOM是数据的一种自然映射。为了提高效率,有些组件会采用数据驱动,即,采用(Model View View Model,MVVM)架构,对DOM做一层封装,当数据发生改变会通知指令去修改对应的DOM,数据驱动DOM变化,提供对View和View Model的双向数据绑定,这使得View Model的状态改变可以自动传递给View,实现数据双向绑定。实际应用场景中,数据驱动的组件所采用的技术栈类型往往为特定技术栈类型,例如,React,Vue,Anglar,等等。因此,本公开实施例中可以先确定该组件当前所适配的技术栈类型,然后检测该组件当前所适配的技术栈类型是否为特定技术栈类型。如果为特定技术栈类型,则可以确定组件类型为数据驱动类型。否则,则可以确定组件类型不为数据驱动类型。

[0103] 步骤S42、在所述组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,根据所述组件的配置参数将所述组件转换为所述结构化描述信息;所述第二技术栈类型为所述组件当前适配的技术栈类型。

[0104] 实际应用场景中,每种技术栈类型都有相应的根据组件的配置参数将组件转换为AST文件的工具。其中,转换工具本质上可以为代码程序。示例的,Vue技术栈类型的组件可以采用“@vue/compiler-sfc”转换工具将组件分解为AST文件,即,实现转换。进一步地,结构化描述信息可以携带组件中的可配置参数。根据组件的配置参数将组件转换为结构化描述信息的过程中,可以基于组件的代码内容以及其中包含的可配置参数,对可配置参数进行划分,使得组件中的各种可配置参数,被划分至对应的预设配置参数类别中,结构化描述信息中可以携带各配置参数类别中的可配置参数。其中,预设配置参数类别可以是根据实际需求设置的,各预设配置参数类别可以包括“样式”、“一般属性”、“计算属性”、“监控属性”、“事件”以及“函数”。或者,也可以采用通用的转换方式进行转换,例如,获取组件的代码中涉及到的各种语法结构以及各语法结构中携带的可配置参数,将各语法结构中携带的

可配置参数划分至预设配置参数类别,并将各个语法结构作为节点,以树状形式记录,进而得到结构化描述信息。进一步地,第二技术栈类型可以是该组件开发时候是采用的技术栈类型,具体类型可以由用户的实际需求决定。

[0105] 本公开实施例中,通过检测组件的组件类型是否为数据驱动类型,在组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,根据所述组件的配置参数将组件转换为结构化描述信息;第二技术栈类型为所述组件当前适配的技术栈类型。这样,以组件当前适配的第二技术栈类型对应的转换工具,直接将组件转换为结构化描述信息,一定程度上可以确保获取结构化描述信息的效率。

[0106] 可选的,本公开实施例中还可以在检测所述组件的组件类型是否为数据驱动类型之后,在检测出所述组件类型不为数据驱动类型的情况下,对于所述组件进行组件类型改造,以将所述组件的组件类型修改为所述数据驱动类型。示例的,可以根据预先设定的指定规则将该组件改造为数据驱动类型的组件,即,在自定义模式下对组件进行分解。其中,指定规则的具体内容可以根据实际需求设置,具体的,可以在该组件的技术栈类型不属于数据驱动类型覆盖的特定技术栈类型的情况下,对该组件进行改造,以使改造后的组件可以适配于属于数据驱动类型覆盖的某个特定技术栈类型,从而方便基于该特定技术栈类型对应的转换工具进行AST文件转换。示例的,对于非数据驱动类型的组件,例如,Jquery组件,指定规则可以为将该组件的初始化阶段传入的参数移动至“一般属性”,以实现数据化改造。其中,初始化阶段传入的参数可以是根据实际需求预先设定的,示例的,初始化阶段传入的参数可以包括“name:UserName;job:UserJob”。在一种现有实现方式中,组件代码中往往仅是通过预设方法定义初始化阶段传入的参数。本公开实施例中,在进行改造时,可以先为该非数据驱动类型的组件添加标签,以通过预设的定义语句,将组件的初始化阶段传入的参数定义至一般属性中。进一步地,为了确保改造后组件的正常使用,可以进一步地为该组件设置特定属性配置(例如,less scoped)、扩展所需继承的对象(例如,base view)以及为该组件定义生命周期函数以及更新函数。其中,生命周期函数可以为mounted函数,更新函数可以为updated函数。

[0107] 当然,也可以采用其他方式实现改造,针对不同组件设置的指定规则可以不同,本公开实施例对此不作限定。本公开实施例中,由于非数据驱动类型的组件往往无法基于转换工具进行转化,因此,通过在组件类型不为数据驱动类型的情况下,对于组件进行组件类型改造,以将组件的组件类型修改为数据驱动类型。在将其修改为数据驱动类型之后,才执行将组件转换为结构化描述信息的操作。这样,可以避免由于对组件执行无用的转换操作,进而浪费处理资源的问题,同时,一定程度上可以提高转换操作的成功率。

[0108] 示例的,图2是本公开实施例提供的一种存储过程示意图,如图2所示,在组件开发阶段,用户可以基于第二技术栈类型开发组件,其中,第二技术栈类型可以是用户根据实际需求任选的。接着,可以通过上传组件的环节,将该组件上传给组件服务端。相应地,组件服务端可以执行检测是否为数据驱动模式组件的操作,即,检测该组件的组件类型是否为数据驱动类型。进一步地,如果是,则可以在第二技术栈类型为VUE类型的情况下,基于VUE类型对应的转换工具分解AST文件;在第二技术栈类型为REACT类型的情况下,基于REACT类型对应的转换工具转换AST文件;在第二技术栈类型为Angular类型的情况下,基于Angular类型对应的转换工具转换AST文件。如果不是,则可以基于自定义模式转换AST文件。

[0109] 进一步地,可以基于组件的AST文件,生成组件抽象化描述信息。具体的,可以在组件存在组件依赖的情况下,将该组件所依赖的组件以及该AST文件作为组件抽象化描述信息。在组件不存在组件依赖的情况下,将该AST文件作为组件抽象化描述信息。最后,可以执行存储操作,以将该组件的标识信息与该组件抽象化描述信息对应存储至组件库。

[0110] 可选的,上述根据目标标识信息查询组件库,从组件库中获取与目标标识信息对应的组件抽象化描述信息的操作,可以具体包括:

[0111] 步骤S52、从所述组件库中各所述组件的标识信息中,查找与所述目标标识信息相匹配的标识信息。

[0112] 本步骤中,可以将目标标识信息与映射关系中各组件的标识信息进行一一对比,以确定两者是否相匹配。其中,与目标标识信息相匹配可以是标识信息中的所有字段内容均匹配,也可以是部分字段内容相匹配,本公开实施例对此不作限定。

[0113] 步骤S53、将所述相匹配的标识信息对应的组件抽象化描述信息,确定为所述目标标识信息对应的组件抽象化描述信息。

[0114] 本步骤中,可以根据映射关系,确定该相匹配的标识信息对应的组件抽象化描述信息,进而得到目标标识信息对应的组件抽象化描述信息,即,目标抽象化描述信息。

[0115] 进一步在,在一种实现方式中,根据第一技术栈类型,对目标组件抽象化描述信息进行标准化代码转化时,可以是将目标抽象化描述信息中的AST文件拆分成若干条单路径代码,然后将单路径转化为指定格式(例如,SSA的格式),之后将SSA格式的单路径封装成符合第一技术栈类型规定的代码语法规范的类,以作为输出。最后将得到的输出以及目标组件抽象化描述信息中的运行依赖,即,所依赖的组件,以组件工程的形式作为标准组件代码,返回给组件获取端。这样,组件获取端拿到的标准组件代码可以是一个标准的基于目标组件抽象化描述信息转换得到的组件工程(包含若干标准文件夹和标准文件)。

[0116] 在一种应用场景中,前端组件可以包括基础组件以及业务组件,基础组件可以具备基础的功能,比如,按钮,输入框,下拉框,等等。基础组件的复用率虽然较高,但是由于业务属性不足,往往需要大量的组件组合才能形成一个具有某种业务功能的组件,例如,登录组件就需要配置了联动关系的至少两个输入框组件以及一个按钮组件。而直接具备业务功能的业务组件,往往会因为存在技术栈类型不通用,例如,不同技术栈类型的组件在不同运行端上可能存在无法通用的情况,例如,在网页(web)端、React Native(RN)端、应用程序端以及小程序端,可能会出现相互隔离无法通用的问题,进而会导致组件复用率较低。现有方式中的组件服务端仅仅是对组件本身进行收录,然后提供查询功能。当用户需要某种功能的组件,通过关键字去搜索,然后从搜索到的结果中选择自己需要的组件。这种方式中,搜索到的组件可能由于技术栈类型不对口无法使用,组件功能不适配,需要进而二次改造,例如,往往需要人工阅读该组件的组件代码,以理解组件代码的编写思路,确定二次改造方式,因此会导致成本较大,进而使得组件复用率较高。

[0117] 示例的,图3是一种现有方式的组件获取过程示意图,如图3所示,用户可以登录组件库,然后通过用户搜索环节,基于关键字从该组件库中搜索所需的组件。接着从搜索到的多个结果中进行选择。例如,用户可以选择结果中的组件C,并对组件C进行改造,以得到适配所需的技术栈类型的组件。进一步地,将改造后的组件直接重新上传至组件库中。现有方式中,由于仍旧是将组件本身直接重新上传至组件库,因此,一定程度上会导致后续依旧出

现搜索到的结果没有满足需求的组件,需要二次改造的问题,从而导致组件获取操作进入不良循环。

[0118] 本公开实施例中,通过将组件的组件抽象化描述信息存储至组件服务端中,使得后续可以基于该组件的组件抽象化描述信息,生成更为标准的适配用户所需使用的技术栈类型的组件并返回给用户,进而一定程度上可以避免由于技术栈类型不匹配,导致需要二次改造,进而导致成本较大,组件复用率较低的问题。需要说明的是,本公开实施例中,组件获取端的用户对获取到的组件进行改进之后,可以将改进后的组件继续存储至组件服务端。相应地,存储的具体实现方式可以参照前述存储过程。相较于直接将二次改造后的组件代码上传至组件服务端的方式,这样,可以避免将二次改造后的组件直接存储至组件服务端,导致检索到的组件无法直接使用,复用率较低的问题。进一步地,本公开实施例还可以向组件获取端,确认用户是否进行配置。如果需要,那么可以通过组件获取端访问组件服务端提供的配置页面,基于该配置页面进行schema配置,例如,进行属性配置,样式配置等基本配置,由于基本配置可以直接在组件服务端内配置完成,因此,一定程度上可以降低使用难度,方便编程能力较弱的用户进行使用。在完成配置之后,再执行将目标抽象化描述信息转换为标准组件代码的操作。

[0119] 示例的,图4是本公开实施例提供的一种组件获取过程示意图,如图4所示,用户可以登录组件库,然后通过用户搜索环节,输入待检索组件的目标标识信息,通过选择技术栈环节,选择待检索组件所需适配的第一技术栈类型,以获取待检索组件对应的目标抽象化描述信息。需要说明的是,组件服务端可以从组件库中筛选与目标标识信息相匹配的组件的组件抽象化描述信息,以作为备选信息。然后,直接从中任选一个备选信息作为目标抽象化描述信息。或者,在选择第一技术栈类型之后,选择对应的第二技术栈类型与第一技术栈类型相适配的备选信息,作为目标抽象化描述信息,以方便后续转化。如果不存在对应的第二技术栈类型与第一技术栈类型相适配的备选信息,则任选一个备选信息作为目标抽象化描述信息。

[0120] 接着,组件服务端可以确认用户是否进行配置,如果需要,则可以在组件服务端进行Schema配置。其中,该组件服务端又可以称为市场端或者组件市场,等等。进一步地,可以在完成配置之后,执行将目标抽象化描述信息转换为标准组件代码的操作。如果不需要进行Schema配置,则可以直接执行将目标抽象化描述信息转换为标准组件代码的操作。相较于上述现有方式中的组件获取过程,本公开实施例中可以直接获取到符合用户所需技术栈类型的标准组件代码,进而使得用户无需进行二次改造即可直接使用,从而可以降低使用成本,提高组件的复用率。

[0121] 需要说明的是,本公开实施例还可以提供一种组件处理方法,该组件处理方法可以应用于用户端,该方法可以包括:根据待存储的各组件的配置参数生成各组件对应的组件抽象化描述信息;将组件抽象化描述信息发送至组件服务端,以便于所述组件服务端获取所述组件的标识信息,并将所述标识信息以及所述组件抽象化描述信息,对应存储至组件服务端。进一步地,还可以提供一种组件处理系统,该组件系统中可以包括上述用户端以及上述组件服务端。进一步地,该系统中还可以包括组件获取端。

[0122] 图5是本公开实施例提供的一种组件处理装置的框图,该装置应用于组件服务端,如图5所示,该装置20可以包括:

[0123] 接收模块201,被配置为接收组件获取端发送的组件获取请求;所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型;

[0124] 第一获取模块202,被配置为根据所述目标标识信息查询组件库,从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息,得到目标抽象化描述信息;其中,所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系;所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息;

[0125] 转化模块203,被配置为根据所述第一技术栈类型,对所述目标抽象化描述信息进行标准化代码转化,生成适配所述第一技术栈类型的标准组件代码;

[0126] 返回模块204,被配置为将所述标准组件代码返回给所述组件获取端。

[0127] 本公开实施例提供的组件处理装置,可以接收组件获取端发送的组件获取请求,组件获取请求中包括待检索组件的目标标识信息以及待检索组件所需适配的第一技术栈类型。然后,根据目标标识信息查询组件库,从组件库中获取与目标标识信息对应的组件抽象化描述信息,得到目标抽象化描述信息。其中,组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系,组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息。根据第一技术栈类型,对目标抽象化描述信息进行标准化代码转化,生成适配第一技术栈类型的标准组件代码,将标准组件代码返回给组件获取端。这样,通过向组件获取端返回适配第一技术栈类型的标准组件代码,可以避免由于技术栈类型不匹配,导致需要二次改造,进而导致成本较大,组件复用率较低的问题。

[0128] 可选的,所述装置20还包括:

[0129] 第二获取模块,被配置为根据各所述组件的配置参数,获取各所述组件对应的组件抽象化描述信息,以及,获取各所述组件的标识信息;

[0130] 存储模块,被配置为将各所述组件的标识信息与所述组件抽象化描述信息对应存储至所述组件库,以生成所述映射关系。

[0131] 可选的,所述第一获取模块202,被具体配置为:

[0132] 从所述组件库中各所述组件的标识信息中,查找与所述目标标识信息相匹配的标识信息;

[0133] 将所述相匹配的标识信息对应的组件抽象化描述信息,确定为所述目标标识信息对应的组件抽象化描述信息。

[0134] 可选的,所述第二获取模块,具体被配置为:

[0135] 对于任一所述组件,检测所述组件是否存在组件依赖,以及,根据所述组件的配置参数获取所述组件对应的结构化描述信息;

[0136] 若所述组件存在组件依赖,则获取所述组件所依赖的组件;将所述结构化描述信息以及所述所依赖的组件,确定为所述组件抽象化描述信息;

[0137] 若所述组件不存在组件依赖,则直接将所述结构化描述信息确定为所述组件抽象化描述信息。

[0138] 可选的,所述第二获取模块,还具体被配置为:

[0139] 检测所述组件的组件类型是否为数据驱动类型;

[0140] 在所述组件类型为数据驱动类型的情况下,以第二技术栈类型对应的转换工具,

根据所述组件的配置参数将所述组件转换为所述结构化描述信息;所述第二技术栈类型为所述组件当前适配的技术栈类型。

[0141] 可选的,所述装置20还包括:

[0142] 改造模块,被配置为在所述组件类型不为数据驱动类型的情况下,对于所述组件进行组件类型改造,以将所述组件的组件类型修改为所述数据驱动类型。

[0143] 可选的,第二获取模块,具体被配置为:

[0144] 接收用户端发送的根据各所述组件的配置参数生成的组件抽象化描述信息;

[0145] 接收所述用户端发送的各所述组件的标识信息。

[0146] 可选的,所述组件标识信息包括组件名称以及所述组件的参数信息。

[0147] 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,此处将不做详细阐述说明。

[0148] 根据本公开的一个实施例,提供了一种电子设备,包括:处理器、用于存储处理器可执行指令的存储器,其中,处理器被配置为执行时实现如上述任一个实施例中的组件处理方法中的步骤。

[0149] 根据本公开的一个实施例,还提供了一种存储介质,当存储介质中的指令由电子设备的处理器执行时,使得电子设备能够执行如上述任一个实施例中的组件处理方法中的步骤。

[0150] 根据本公开的一个实施例,还提供了一种计算机程序产品,该计算机程序产品包括可读性程序指令,可读性程序指令由电子设备的处理器执行时,使得电子设备能够执行如上述任一个实施例中的组件处理方法中的步骤。

[0151] 图6是根据一示例性实施例示出的一种用于组件处理的装置的框图。例如,装置700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。

[0152] 参照图6,装置700可以包括以下一个或多个组件:处理组件702,存储器704,电力组件706,多媒体组件708,音频组件710,输入/输出(I/O)的接口712,传感器组件714,以及通信组件716。

[0153] 处理组件702通常控制装置700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702可以包括一个或多个处理器720来执行指令,以完成上述的组件处理方法的全部或部分步骤。此外,处理组件702可以包括一个或多个模块,便于处理组件702和其他组件之间的交互。例如,处理组件702可以包括多媒体模块,以方便多媒体组件708和处理组件702之间的交互。

[0154] 存储器704被配置为存储各种类型的数据以支持在装置700的操作。这些数据的示例包括用于在装置700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704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。

[0155] 电源组件706为装置700的各种组件提供电力。电源组件706可以包括电源管理系统,一个或多个电源,及其他与为装置700生成、管理和分配电力相关联的组件。

[0156] 多媒体组件708包括在所述装置700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708包括一个前置摄像头和/或后置摄像头。当装置700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。

[0157] 音频组件710被配置为输出和/或输入音频信号。例如,音频组件710包括一个麦克风(MIC),当装置700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704或经由通信组件716发送。在一些实施例中,音频组件710还包括一个扬声器,用于输出音频信号。

[0158] I/O接口712为处理组件702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。

[0159] 传感器组件714包括一个或多个传感器,用于为装置700提供各个方面的状态评估。例如,传感器组件714可以检测到装置700的打开/关闭状态,组件的相对定位,例如所述组件为装置700的显示器和小键盘,传感器组件714还可以检测装置700或装置700一个组件的位置改变,用户与装置700接触的存在或不存在,装置700方位或加速/减速和装置700的温度变化。传感器组件714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714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714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。

[0160] 通信组件716被配置为便于装置700和其他设备之间有线或无线方式的通信。装置700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716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716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。

[0161] 在示例性实施例中,装置700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,用于执行上述组件处理方法。

[0162] 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704,上述指令可由装置700的处理器720执行以完成上述组件处理方法。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。

[0163] 图7是根据一示例性实施例示出的一种用于组件处理的装置的框图。例如,装置800可以被提供为一服务器。参照图7,装置800包括处理组件822,其进一步包括一个或多个处理器,以及由存储器832所代表的存储器资源,用于存储可由处理组件822的执行的指令,

例如应用程序。存储器832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822被配置为执行指令,以执行上述组件处理方法。

[0164] 装置800还可以包括一个电源组件826被配置为执行装置800的电源管理,一个有线或无线网络接口850被配置为将装置800连接到网络,和一个输入输出(I/O)接口858。装置800可以操作基于存储在存储器832的操作系统,例如Windows ServerTM,Mac OS XTM,UnixTM,LinuxTM,FreeBSDTM或类似。

[0165] 本领域技术人员在考虑说明书及实践这里公开的发明后,将容易想到本公开的其它实施方案。本申请旨在涵盖本公开的任何变型、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公开的一般性原理并包括本公开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,本公开的真正范围和精神由下面的权利要求指出。

[0166] 应当理解的是,本公开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。本公开的范围仅由所附的权利要求来限制。

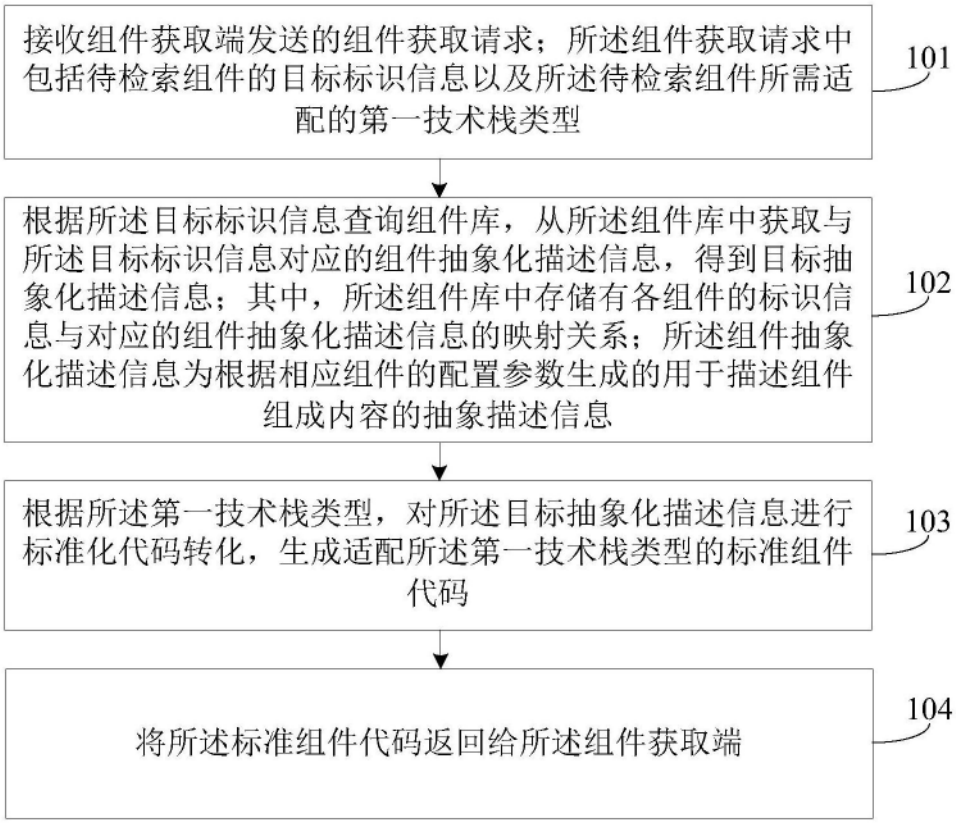

接收组件获取端发送的组件获取请求；所述组件获取请求中包括待检索组件的目标标识信息以及所述待检索组件所需适配的第一技术栈类型
101
根据所述目标标识信息查询组件库，从所述组件库中获取与所述目标标识信息对应的组件抽象化描述信息，得到目标抽象化描述信息；其中，所述组件库中存储有各组件的标识信息与对应的组件抽象化描述信息的映射关系；所述组件抽象化描述信息为根据相应组件的配置参数生成的用于描述组件组成内容的抽象描述信息
102
根据所述第一技术栈类型，对所述目标抽象化描述信息进行标准化代码转化，生成适配所述第一技术栈类型的标准组件代码
103
将所述标准组件代码返回给所述组件获取端
104

图1

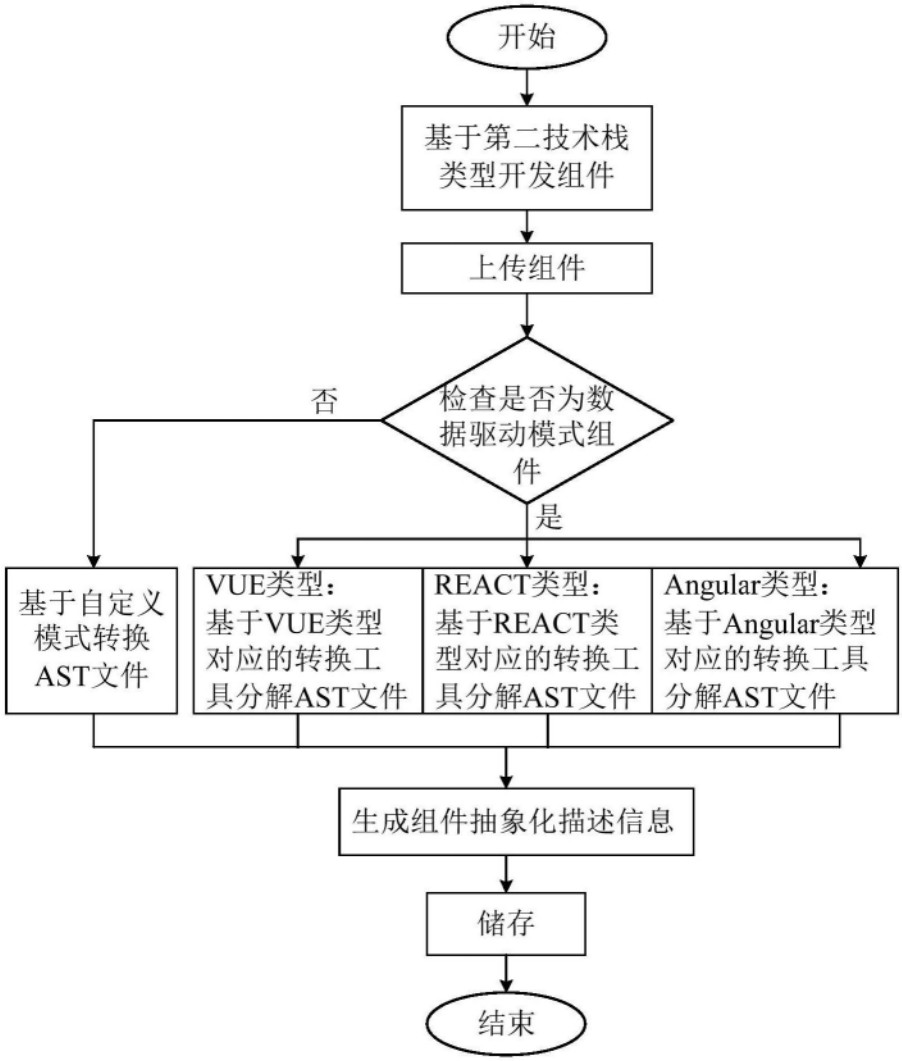

开始
基于第二技术栈类型开发组件
上传组件
检查是否为数据驱动模式组件
否
是
基于自定义模式转换AST文件
VUE类型：基于VUE类型对应的转换工具分解AST文件
REACT类型：基于REACT类型对应的转换工具分解AST文件
Angular类型：基于Angular类型对应的转换工具分解AST文件
生成组件抽象化描述信息
储存
结束

图2

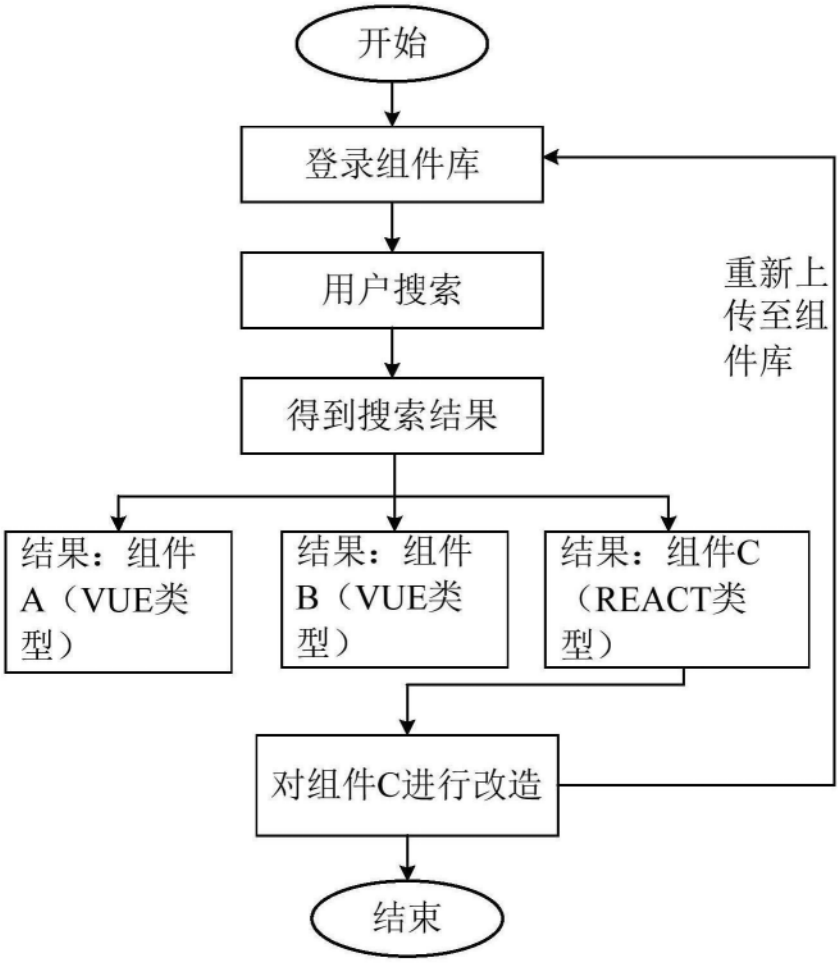

开始
登录组件库
用户搜索
得到搜索结果
结果：组件A（VUE类型）
结果：组件B（VUE类型）
结果：组件C（REACT类型）
对组件C进行改造
结束
重新上传至组件库

图3

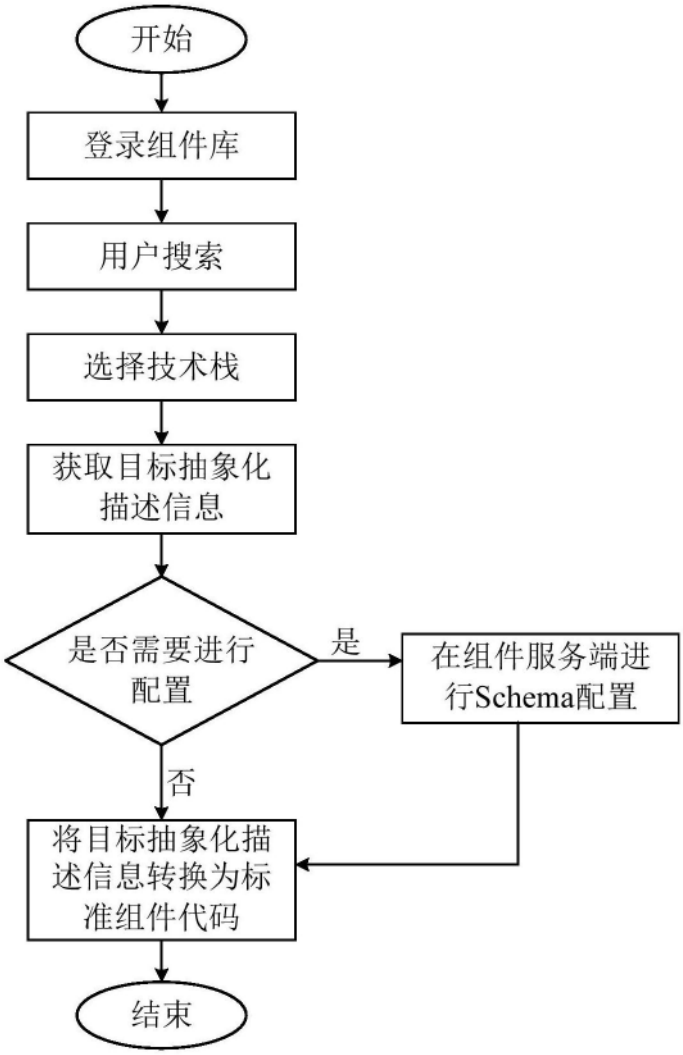

开始
登录组件库
用户搜索
选择技术栈
获取目标抽象化描述信息
是否需要进行配置
是
在组件服务端进行Schema配置
否
将目标抽象化描述信息转换为标准组件代码
结束

图4

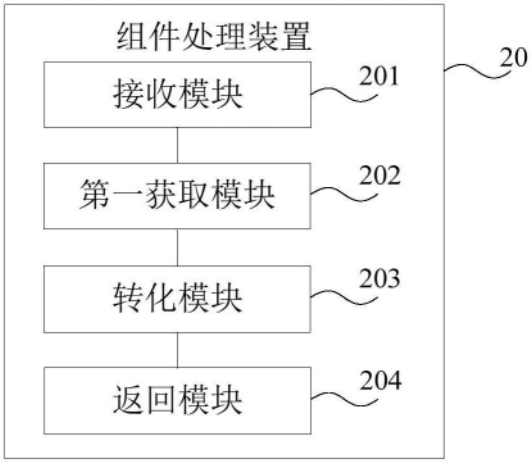

组件处理装置
20
接收模块
201
第一获取模块
202
转化模块
203
返回模块
204

图5

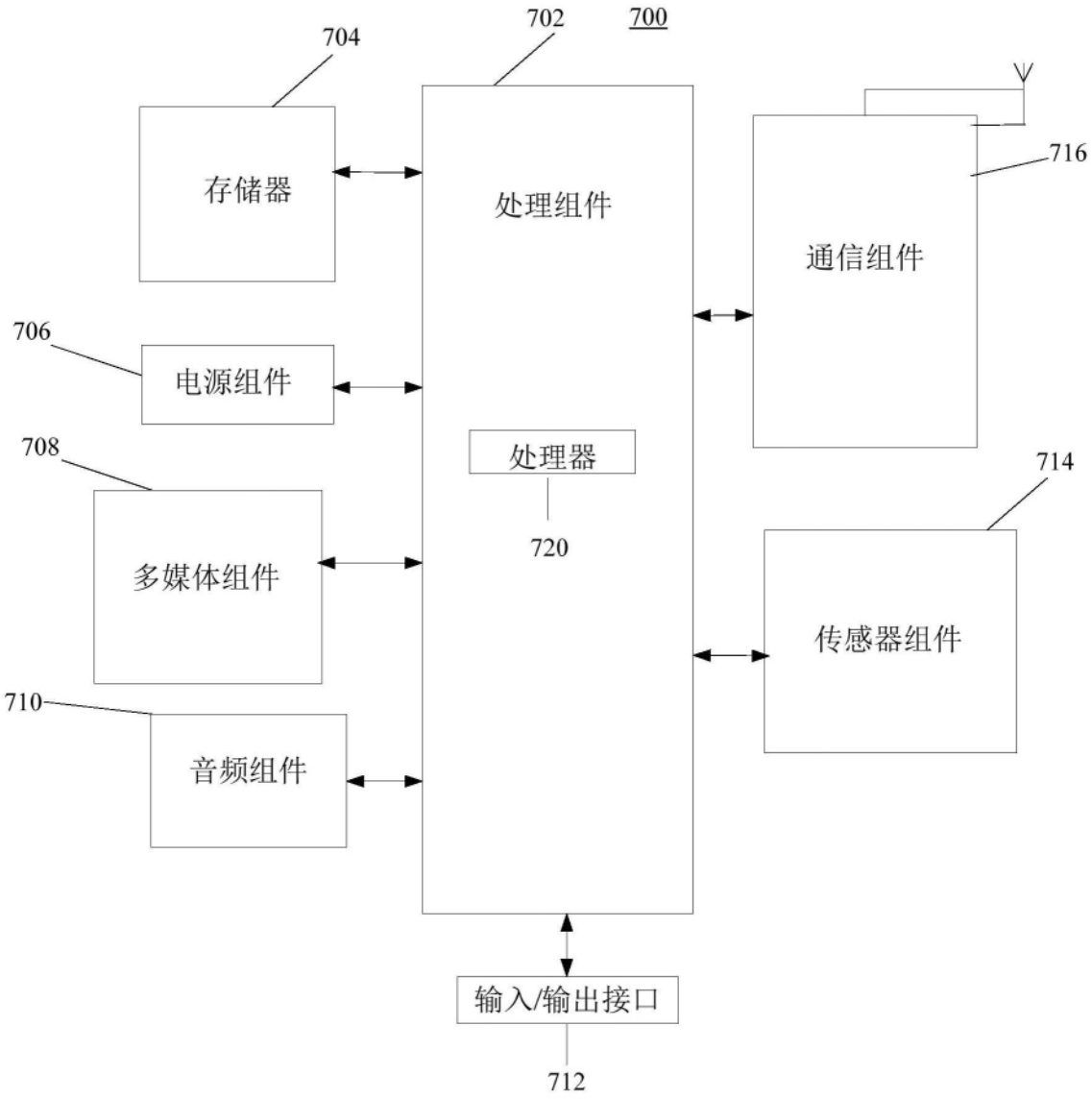

704
702
700
存储器
处理组件
716
通信组件
706
电源组件
处理器
720
708
多媒体组件
714
传感器组件
710
音频组件
输入/输出接口
712

图6

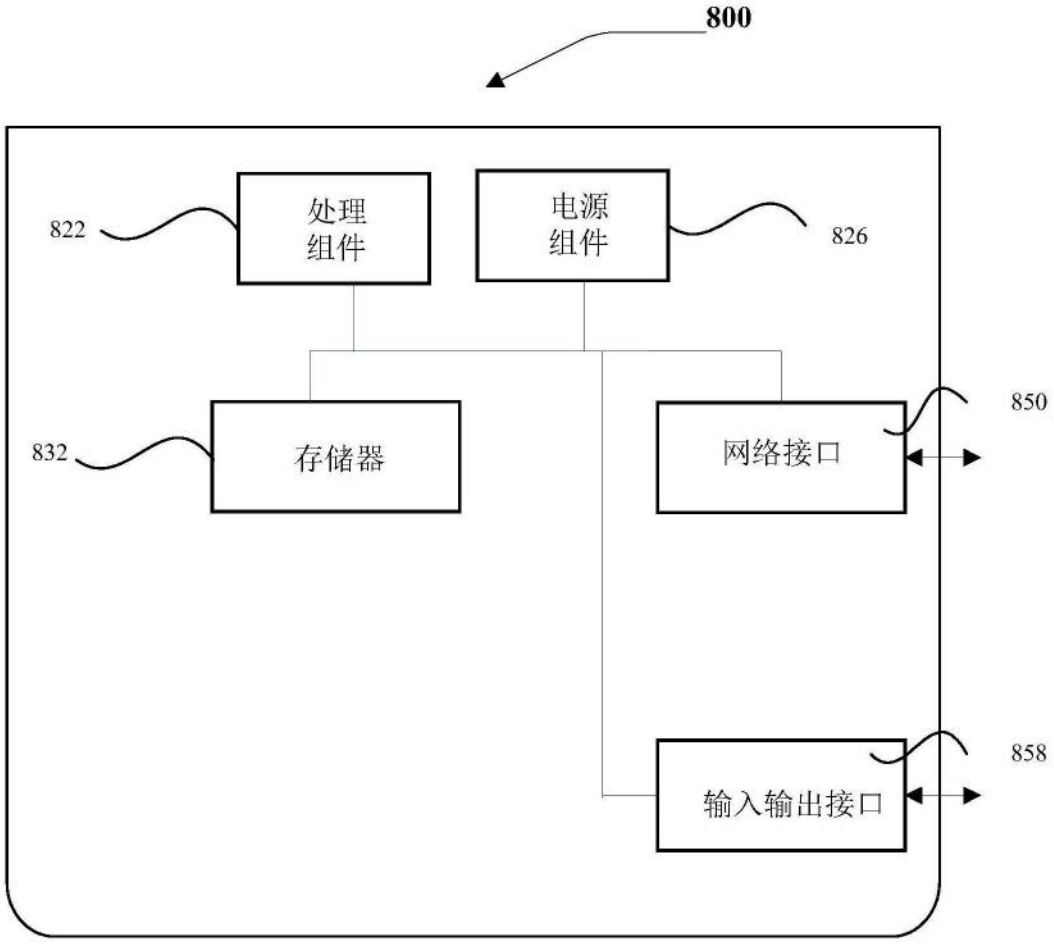


图7