(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 110716866 B
(45)授权公告日 2024.06.28

(21)申请号 201910843704.8

(22)申请日 2019.09.06

(65)同一申请的已公布的文献号
申请公布号 CN 110716866 A

(43)申请公布日 2020.01.21

(73)专利权人 中国平安财产保险股份有限公司
地址 518000 广东省深圳市福田区益田路5033号平安金融中心12、13、38、39、40层

(72)发明人 傅女婷

(74)专利代理机构 深圳市力道知识产权代理事务所(普通合伙) 44507
专利代理师 何姣

(51)Int.Cl.
G06F 11/36(2006.01)

(56)对比文件
CN 105320591 A,2016.02.10
CN 109032949 A,2018.12.18

审查员 邹玥

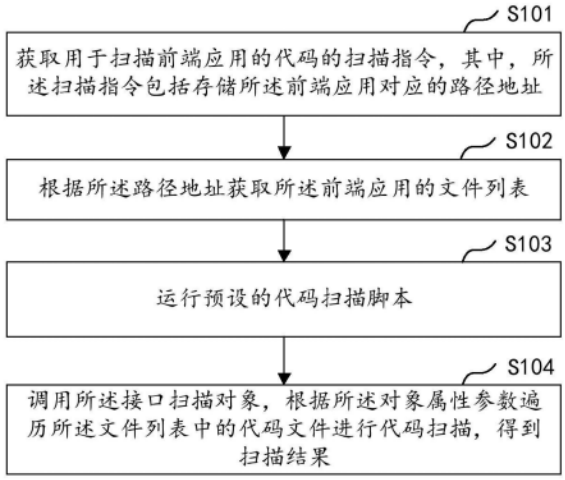
权利要求书2页 说明书11页 附图6页

(54)发明名称

代码质量扫描方法、装置、计算机设备及存储介质

(57)摘要

本申请涉及APP功能测试,具体公开了一种代码质量扫描方法、装置、设备及存储介质,其中方法包括:获取用于扫描前端应用的代码的扫描指令,其中,所述扫描指令包括存储所述前端应用对应的路径地址;根据所述路径地址获取所述前端应用的文件列表;运行预设的代码扫描脚本,其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数;调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。该方法可以提高前端应用的代码扫描效率以及代码质量标准。



CN 110716866 B

1.一种代码质量扫描方法,其特征在于,应用于服务器,包括:

接收终端发送的用于扫描前端应用的代码的扫描指令,其中,所述扫描指令为所述终端根据用户输入的前端应用的应用标识和路径地址生成的扫描指令;

接收其余服务器发送的所述前端应用的文件列表和代码文件;其中,所述其余服务器根据所述路径地址中获取所述前端应用的各个代码文件,以及根据所述路径地址中路径关键字对各个代码文件进行分类,并将分类后的代码文件对应的文件名称记录在预设列表中,得到所述前端应用的文件列表;

运行预设的代码扫描脚本,其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数;

调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果;

其中,所述对象属性参数通过以下方式得到:

获取预先创建的代码扫描脚本,在所述代码扫描脚本中添加ESLint代码检测工具;

选择所述ESLint代码检测工具中的CLIEngine对象作为接口扫描对象;

根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数;

其中,所述根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数,包括:

获取用户预先制定的扫描规则文本,将所述扫描规则文本转换成多种格式的配置文件;

将多种格式的配置文件保存至所述ESLint代码检测工具的特定目录生成配置目录;

从所述配置目录中读取相应的配置文件对所述接口扫描对象进行配置,得到对象属性参数。

2.根据权利要求1所述的代码质量扫描方法,其特征在于,所述获取用户预先制定的扫描规则文本,包括:

显示规则显示界面以及在所述规则显示界面中显示默认扫描规则文本;

监测用户是否对所述默认扫描规则文本进行修改操作;

若用户对所述默认扫描规则文本进行修改操作,则保存并获取修改后的默认扫描规则文本作为预先制定的扫描规则文本。

3.根据权利要求1所述的代码质量扫描方法,其特征在于,还包括:

根据所述扫描结果生成扫描报告,并将所述扫描报告发送至用户。

4.一种代码质量扫描装置,其特征在于,包括:

指令获取模块,用于接收终端发送的用于扫描前端应用的代码的扫描指令,其中,所述扫描指令为所述终端根据用户输入的前端应用的应用标识和路径地址生成的扫描指令;

列表获取模块,用于接收其余服务器发送的所述前端应用的文件列表和代码文件;其中,所述其余服务器根据所述路径地址中获取所述前端应用的各个代码文件,以及根据所述路径地址中路径关键字对各个代码文件进行分类,并将分类后的代码文件对应的文件名称记录在预设列表中,得到所述前端应用的文件列表;

脚本运行模块,用于运行预设的代码扫描脚本,其中,所述代码扫描脚本中添加有用于

创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数;

调用扫描模块,用于调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果;

其中,所述对象属性参数通过以下方式得到:

获取预先创建的代码扫描脚本,在所述代码扫描脚本中添加ESLint代码检测工具;

选择所述ESLint代码检测工具中的CLIEngine对象作为接口扫描对象;

根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数;

其中,所述根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数,包括:

获取用户预先制定的扫描规则文本,将所述扫描规则文本转换成多种格式的配置文件;

将多种格式的配置文件保存至所述ESLint代码检测工具的特定目录生成配置目录;

从所述配置目录中读取相应的配置文件对所述接口扫描对象进行配置,得到对象属性参数。

5.一种计算机设备,其特征在于,所述计算机设备包括存储器和处理器;

所述存储器用于存储计算机程序;

所述处理器,用于执行所述计算机程序并在执行所述计算机程序时实现如权利要求1至3中任一项所述的代码质量扫描方法。

6.一种计算机可读存储介质,其特征在于,所述计算机可读存储介质存储有计算机程序,所述计算机程序被处理器执行时使所述处理器实现如权利要求1至3中任一项所述的代码质量扫描方法。

代码质量扫描方法、装置、计算机设备及存储介质

技术领域

[0001] 本申请涉及互联网技术领域，尤其涉及一种代码质量扫描方法、装置、计算机设备及存储介质。

背景技术

[0002] 目前，代码质量扫描作为前端应用的安全解决方案，是指研发人员写好代码后，无需经过编译器编译，而直接使用一些扫描工具对其进行扫描，识别出代码中存在的一些质量问题，比如存在的一些语义缺陷、安全漏洞等等。然而目前，对每个前端应用进行代码扫描，均需要安装代码扫描工具以及制定相应的代码扫描规则，由此导致了大量的重复性工作，浪费了时间和人力，同时降低了代码扫描效率。

发明内容

[0003] 本申请提供了一种代码质量扫描方法、装置、计算机设备及存储介质，以在。

[0004] 第一方面，本申请提供了一种代码质量扫描方法，所述方法包括：

[0005] 获取用于扫描前端应用的代码的扫描指令，其中，所述扫描指令包括存储所述前端应用对应的路径地址；

[0006] 根据所述路径地址获取所述前端应用的文件列表；

[0007] 运行预设的代码扫描脚本，其中，所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数；

[0008] 调用所述接口扫描对象，根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描，得到扫描结果。

[0009] 第二方面，本申请还提供了一种代码质量扫描装置，所述装置包括：

[0010] 指令获取模块，用于获取用于扫描前端应用的代码的扫描指令，其中，所述扫描指令包括存储所述前端应用对应的路径地址；

[0011] 列表获取模块，用于根据所述路径地址获取所述前端应用的文件列表；

[0012] 脚本运行模块，用于运行预设的代码扫描脚本，其中，所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数；

[0013] 调用扫描模块，用于调用所述接口扫描对象，根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描，得到扫描结果。

[0014] 第三方面，本申请还提供了一种计算机设备，所述计算机设备包括存储器和处理器；所述存储器用于存储计算机程序；所述处理器，用于执行所述计算机程序并在执行所述计算机程序时实现如上述的代码质量扫描方法。

[0015] 第四方面，本申请还提供了一种计算机可读存储介质，所述计算机可读存储介质存储有计算机程序，所述计算机程序被处理器执行时使所述处理器实现如上述的代码质量

扫描方法。

[0016] 本申请公开了一种代码质量扫描方法、装置、计算机设备及存储介质，通过扫描指令获取前端应用的路径地址；根据前端应用的路径地址并获取该前端应用的文件列表；运行预先创建的代码扫描脚本，调用接口扫描对象，根据对象属性参数遍历文件列表中代码文件实现代码质量扫描。该方法可以针对不同的前端应用进行代码扫描，无需针对每个前端应用均安装代码检测工具以及制定相应的代码扫描规则，因此提高了前端应用的代码扫描效率，同时利用统一的代码扫描规则，保证代码质量标准，以便后续维护修改。

附图说明

[0017] 为了更清楚地说明本申请实施例技术方案，下面将对实施例描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0018] 图1是本申请的实施例提供的一种代码质量扫描方法的示意流程图；

[0019] 图2是本申请的实施例提供的另一种代码质量扫描方法的示意流程图；

[0020] 图3a是本申请的实施例提供的代码质量扫描方法的应用场景示意图；

[0021] 图3b是本申请的实施例提供的又一种代码质量扫描方法的示意流程图；

[0022] 图4为本申请的实施例提供的一种代码质量扫描装置的示意性框图；

[0023] 图5为本申请的实施例提供的另一种代码质量扫描装置的示意性框图；

[0024] 图6为本申请的实施例提供的一种计算机设备的结构示意性框图。

具体实施方式

[0025] 下面将结合本申请实施例中的附图，对本申请实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例是本申请一部分实施例，而不是全部的实施例。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

[0026] 附图中所示的流程图仅是示例说明，不是必须包括所有的内容和操作/步骤，也不是必须按所描述的顺序执行。例如，有的操作/步骤还可以分解、组合或部分合并，因此实际执行的顺序有可能根据实际情况改变。

[0027] 应当理解，在此本申请说明书中所使用的术语仅仅是出于描述特定实施例的目的而并不意在限制本申请。如在本申请说明书和所附权利要求书中所使用的那样，除非上下文清楚地指明其它情况，否则单数形式的“一”、“一个”及“该”意在包括复数形式。

[0028] 还应当进理解，在本申请说明书和所附权利要求书中使用的术语“和/或”是指相关联列出的项中的一个或多个的任何组合以及所有可能组合，并且包括这些组合。

[0029] 本申请的实施例提供了一种代码质量扫描方法、装置、计算机设备及存储介质。其中，该代码质量扫描方法可以应用于服务器中，对开发的各种前端应用的代码文件进行扫描以确定相应的代码质量。其中，该服务器可以为独立的服务器，也可以为服务器集群。

[0030] 下面结合附图，对本申请的一些实施方式作详细说明。在不冲突的情况下，下述的实施例及实施例中的特征可以相互组合。

[0031] 请参阅图1，图1是本申请的实施例提供的一种代码质量扫描方法的示意流程图。

该代码质量扫描方法可以快速确定前端应用的代码质量,比如是否存在语义缺陷以及安全漏洞等问题。

[0032] 如图1所示,该代码质量扫描方法具体包括步骤S101至步骤S104。

[0033] S101、获取用于扫描前端应用的代码的扫描指令,其中,所述扫描指令包括存储所述前端应用对应的路径地址。

[0034] 具体地,在用户需要对某个前端应用进行代码质量扫描时,即扫描需求需,要提供前端应用的应用标识以及路径地址,所述应用标识可以为应用名称,比如平安好车主,所述路径地址为前端应用的代码文件对应的保存地址,可以为保存在开发该前端应用对应的终端或服务器中对应的地址。

[0035] 具体地,可以是终端根据用户提供的扫描需求生成扫描指令,将生成的扫描指令发送给服务器;也可以是服务器根据扫描需求生成扫描指令。服务器在获取到扫描指令时,解析所述扫描指令以获取所述扫描指令中的前端应用的路径地址。

[0036] S102、根据所述路径地址获取所述前端应用的文件列表。

[0037] 具体地,服务器根据用户通过扫描指令提供的前端应用的路径地址,获取所述前端应用的文件列表,所述文件列表包括前端应用中所有待扫描的代码文件,以便对所有的待扫描的代码文件中的源代码进行扫描。

[0038] S103、运行预设的代码扫描脚本。

[0039] 其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数。所述代码扫描脚本包括通过引入代码检测工具创建的接口扫描对象以及接口扫描对象对应的对象属性参数。在接收到扫描指令后,服务器启动并运行该代码扫描脚本,由此为前端应用的代码扫描提供了基本的运行服务。

[0040] 在一个实施例中,在所述运行预设的代码扫描脚本之前,还包括:获取预先创建的代码扫描脚本,在所述代码扫描脚本中添加ESLint代码检测工具;选择所述ESLint代码检测工具中的CLIEngine对象作为接口扫描对象;根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数。

[0041] 具体地,利用Node.js创建代码扫描脚本,Node.js是运行在服务端的JavaScript,在相对较低的系统资源下也具有较高的性能以及出众的负载能力,适合用作依赖其它IO资源的中间层服务,因此可以利用Node.js开发代码扫描脚本,以便通过运行该代码扫描脚本时为多个前端应用的代码扫描提供的基本的运行服务。

[0042] 在创建完代码扫描脚本后,在该代码扫描脚本中插入代码检测工具,插入的代码检测工具比如为ESLint,所述ESLint是一个插件化的JavaScript代码检测工具,ESLint代码检查是一种静态分析过程,用于寻找有问题的代码,比如存在语义缺陷或安全漏洞的代码,并不依赖于具体的编码风格。

[0043] 由于代码扫描脚本是采用Node.js创建的,而ESLint也是使用Node.js编写,因此便于在代码扫描脚本中插入Elint代码扫描工具,同时二者结合可以提供一个快速的运行环境。

[0044] 其中,根据代码检测工具创建接口扫描对象,是指通过代码检测工具创建一个接口对象,比如引入ESLint的CLIEngine对象作为接口扫描对象。根据预先制定的代码扫描规

则对所述接口扫描对象进行配置,得到对象属性参数,预先制定的代码扫描规则是统一的扫描规则,具体可以一个公司内部制定的统一扫描规则,也可以是一个大的项目下制定的统一扫描规则。具体地,可以基于CLIEngine对象的executeOnFiles或getFormatter函数,根据获取的代码扫描规则对接口扫描对象进行配置,得到对象属性参数,以便在调用该接口扫描对象时利用该对象属性参数进行扫描。

[0045] 在一个实施例中,所述根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数,包括:获取用户预先制定的扫描规则文本,将所述扫描规则文本转换成多种格式的配置文件;将多种格式的配置文件保存至所述ESLint代码检测工具的特定目录生成配置目录;从所述配置目录中读取相应的配置文件对所述接口扫描对象进行配置,得到对象属性参数。

[0046] 具体地,所述扫描规则文本包括相应的代码规范要求,所述代码规范要求具体如下:

[0047] //禁止条件表达式中出现赋值操作符

[0048] "no-cond-assign":2,

[0049] //禁用console

[0050] "no-console":0,

[0051] //禁止在条件中使用常量表达式

[0052] //if(false){

[0053] //doSomethingUnfinished();

[0054] //}//cuowu

[0055] "no-constant-condition":2,

[0056] //禁止在正则表达式中使用控制字符:new RegExp("\x1f")

[0057] "no-control-regex":2,

[0058] //数组和对象键值对最后一个逗号,never参数:不能带末尾的逗号,always参数:必须带末尾的逗号,

[0059] //always-multiline:多行模式必须带逗号,单行模式不能带逗号

[0060] "comma-dangle":[1,"always-multiline"],

[0061] //禁用debugger

[0062] "no-debugger":2,

[0063] //禁止function定义中出现重名参数

[0064] "no-dupe-args":2,

[0065] //禁止对象字面量中出现重复的key

[0066] "no-dupe-keys":2,

[0067] //禁止重复的case标签

[0068] "no-duplicate-case":2,

[0069] //禁止空语句块

[0070] "no-empty":2,

[0071] 将预先制定的扫描规则文本转换成多种格式的配置文件,其中,该多种格式包括:.ESLintrc.js格式、.ESLintrc.yaml格式、.ESLintrc.yml格式、.ESLintrc.json格式和

package.json格式等,将多个不同格式的配置文件均保持在代码检测工具对应的特定目录,该特定目录为引入代码检测工具时生成一个配置目录用于保存配置文件。进而从所述配置目录中读取相应的配置文件对所述接口扫描对象进行配置,得到对象属性参数,该对象属性参数也包括多种格式参数,以便不同的前端应用的代码文件进行扫描。

[0072] 在一个实施例中,所述获取用户预先制定的扫描规则文本,包括:显示规则显示界面以及在所述规则显示界面中显示默认扫描规则文本;监测用户是否对所述默认扫描规则文本进行修改操作;若用户对所述默认扫描规则文本进行修改操作,则保存并获取修改后的默认扫描规则文本作为预先制定的扫描规则文本。

[0073] 具体地,为了让用户可以快速地创建自己的代码扫描规则。在代码扫描工具中内置了一些默认扫描规则文档,当加载代码扫描规则对接口扫描对象进行配置时,显示规则显示界面,该规则显示界面可以为弹框显示界面,以及在所述规则显示界面中显示默认扫描规则文本。监测用户是否对所述默认扫描规则文本进行修改操作,比如添加某些规则内容或删除某些规则内容;若用户对所述默认扫描规则文本进行修改操作,则保存并获取修改后的默认扫描规则文本作为预先制定的扫描规则文本,进而实现了用户快速创建自己的代码扫描规则,以便实现对接口扫描对象进行配置,得到对象属性参数。

[0074] S104、调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0075] 具体地,在启动该并运行该代码扫描脚本后,自动调用代码扫描脚本中的接口扫描对象,并根据所述对象属性参数遍历文件列表中代码文件实现代码扫描,得到扫描结果。其中所述对象属性参数中包括用户统一制定的代码扫描规则,利用该统一的代码扫描规则扫描文件列表中的代码文件,得到扫描结果,该扫描结果可能包括:有语义缺陷、安全漏洞和不符合规定的代码等。

[0076] 上述实施例提供的代码质量扫描方法可以对不同的前端应用进行扫描,无需针对每个前端应用均安装代码检测工具以及制定相应的代码扫描规则,因此该代码质量扫描方法提高了前端应用的代码扫描效率,同时利用统一的代码扫描规则,进而保证代码质量达到同一高水准,以便后续方便维护修改。

[0077] 请参阅图2,图2是本申请的实施例提供的代码质量扫描方法的应用场景示意图。该代码质量扫描方法可以快速确定前端应用的代码质量,比如是否存在语义缺陷以及安全漏洞等问题。

[0078] 如图2所示,该代码质量扫描方法具体包括步骤S201至步骤S206。

[0079] S201、接收终端发送的用于扫描前端应用的代码的扫描指令,其中,所述扫描指令包括存储所述前端应用对应的路径地址。

[0080] 终端在获取到用户提供的前端应用的应用标识以及路径地址,并根据前端应用的应用标识以及路径地址生成扫描指令,并发送该扫描指令至服务端,服务端接收终端发送的用于扫描前端应用的代码的扫描指令。

[0081] 比如,可以在终端中设置一个扫描工具软件,用户需要对某个正在开发或者已经开发完成的前端应用进行代码质量扫描时,其可以打开该扫描工具软件,扫描工具软件可以显示一个信息输入界面,以便用户在该信息输入界面中应用标识以及路径地址后并进确认后,终端通过该扫描工具软件获取前端应用的应用标识以及路径地址,并根据前端应用

的应用标识以及路径地址生成扫描指令。

[0082] S202、根据所述路径地址获取所述前端应用的各个代码文件。

[0083] 具体地,服务器根据所述路径地址查询并获取所述前端应用的各个代码文件,是指获取该前端应用对应的路径地址中涉及的所有代码文件。

[0084] 比如,提供的地址路径为E:\Document\learning\works\project\Item4\,其中,根据所述路径地址获取所述前端应用的各个代码文件,具体是指获取Document、learning、works、roject和Item4等目录下的所有代码文件。

[0085] S203、根据所述路径地址中路径关键字对各个代码文件进行分类,并将分类后的代码文件对应的文件名称记录在预设列表中,得到所述前端应用的文件列表。

[0086] 其中,所述路径地址中路径关键字具体为路径地址中涉及目录名称,比如Document、learning、works、project和Item4均为路径关键字,因为不同的路径关键字对应路径目录下均会包括相应的代码文件,因此根据路径地址中路径关键字对所有代码文件进行分类,并将分类后的代码文件记录在预设列表中以生成文件列表。以便根据文件列表中代码文件的分类关系进行代码扫描,由此提供了扫描速度。

[0087] S204、运行预设的代码扫描脚本。

[0088] 其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数。

[0089] S205、调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0090] 在运行代码扫描脚本后,自动调用代码扫描脚本中的接口扫描对象,并根据所述对象属性参数遍历分类后生成的文件列表中代码文件实现代码扫描,得到扫描结果。其中,所述扫描结果可能包括:有语义缺陷、安全漏洞和不符合规定的代码等。

[0091] S206、根据所述扫描结果生成扫描报告,并将所述扫描报告发送至用户。

[0092] 在得到扫描结果后,对所述扫描结果进行统计分析,比如统计每一个类型代码存在错误的数量等,并根据统计分析对应的统计结果生成扫描报告,在代码扫描脚本中插入Nodemailer,利用插件Nodemailer将扫描报告发送到用户指定邮箱,随时统计前端应用和评估前端应用的开发质量。

[0093] 上述实施例提供的代码质量扫描方法可以对不同的前端应用的代码文件进行快速扫描,无需针对每个前端应用均安装代码检测工具以及制定相应的代码扫描规则,因此该代码质量扫描方法提高了前端应用的代码扫描效率,同时利用统一的代码扫描规则,进而保证代码质量达到同一高水准,以便后续方便维护修改。

[0094] 请参阅图3a和图3b,图3a是本申请的实施例提供的代码质量扫描方法的应用场景示意图;图3b是本申请的实施例提供的另一种代码质量扫描方法的示意流程图。

[0095] 其中,该应用场景包括服务器A、多个服务器N以及终端,其中服务器A与终端、服务器N配合完成代码质量扫描方法,服务器A中预先存储有代码扫描脚本,服务器N中存储有已开发的前端应用的代码文件,终端用于生成扫描指令。

[0096] 需要说明的是,每个服务器N可以存储一个前端应用的代码文件。比如服务器1中存储前端应用1,服务器2中存储有前端应用2,前端应用1和前端应用2是两个完全不同的应用。

[0097] 以下将结合图3a中的应用场景,对本申请的实施例提供的另一种代码质量扫描方法进行详细介绍。如图3b所示,该代码质量扫描方法具体包括步骤S301至步骤S311。

[0098] S301、当检测到用户输入预设指令字符时,输出提示信息。

[0099] 其中,提示信息用于提示用户输入前端应用的应用标识和路径地址,具体可以是语音提示信息或文字提示信息。预设指令字符可以命令字符,比如输入ceshi字符等。

[0100] S302、获取用户输入的前端应用的应用标识和路径地址。

[0101] 具体地,终端获取用户根据所述提示信息输入的前端应用的应用标识和路径地址。

[0102] S303、根据获取到的应用标识和路径地址生成扫描指令。

[0103] 具体地,终端根据获取到的应用标识和路径地址生成扫描指令。该扫描指令用于指令服务器A启动并运行预设的代码扫描脚本,以及用于指令服务器N对代码文件进行分类,N为正整数。

[0104] S304、发送所述扫描指令。

[0105] 其中,所述扫描指令包括存储所述前端应用对应的路径地址。具体地,分别发送扫描指令至服务器A和服务器N。

[0106] S305、接收所述扫描指令,根据所述路径地址获取所述前端应用的各个代码文件。

[0107] 服务器N接收所述扫描指令,并根据所述路径地址获取应用标识对应的前端应用的各个代码文件。

[0108] S306、根据所述路径地址中路径关键字对各个代码文件进行分类。

[0109] 其中,所述路径地址中路径关键字具体为路径地址中涉及目录名称,比如Document、learning、works、project和Item4均为路径关键字,因为不同的路径关键字对应路径目录下均会包括相应的代码文件。由服务器N根据路径地址中路径关键字对所有代码文件进行分类,得到分类后的代码文件。

[0110] S307、将分类后的代码文件对应的文件名称记录在预设列表中,得到所述前端应用的文件列表。

[0111] 具体地,服务器N将分类后的代码文件记录在预设列表中以生成文件列表。

[0112] S308、发送所述文件列表以及所述文件列表中的代码文件。

[0113] 具体地,服务器N发送所述文件列表以及所述文件列表中的代码文件至服务器A。

[0114] S309、接收所述扫描指令,运行预设的代码扫描脚本。

[0115] 其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数。具体地,服务器A接收所述扫描指令,运行预设的代码扫描脚本。

[0116] S310、调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0117] 具体地,服务器A调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0118] S311、根据所述扫描结果生成扫描报告,并将所述扫描报告发送至用户。

[0119] 具体地,服务器A根据所述扫描结果生成扫描报告,并将所述扫描报告发送至用户使用的终端。

[0120] S312、显示所述扫描报告。

[0121] 终端接收服务器A发送的扫描报告,并显示所述扫描报告,以便用户(应用开发人员)查看,了解前端应用的代码存在的质量问题。

[0122] 上述实施例提供的代码质量扫描方法可以实现每个用户对自己开发的前端应用进行扫描,而无需针对每个前端应用均安装代码检测工具以及制定相应的代码扫描规则,同时利用前端应用对应的服务器对代码文件进行分类,进一步地提高了前端应用的代码扫描效率,同时利用统一的代码扫描规则,进而保证代码质量达到同一高水准,以便后续方便维护修改。

[0123] 请参阅图4,图4是本申请的实施例提供一种代码质量扫描装置的示意性框图,该代码质量扫描装置用于执行前述的代码质量扫描方法。其中,该代码质量扫描装置可以配置于服务器。

[0124] 如图6所示,该代码质量扫描装置400,包括:指令获取模块401、列表获取模块402、脚本运行模块403和调用扫描模块404。

[0125] 指令获取模块401,用于获取用于扫描前端应用的代码的扫描指令,其中,所述扫描指令包括存储所述前端应用对应的路径地址。

[0126] 列表获取模块402,用于根据所述路径地址获取所述前端应用的文件列表。

[0127] 脚本运行模块403,用于运行预设的代码扫描脚本,其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数。

[0128] 在一些实施例中,脚本运行模块403,具体用于:获取预先创建的代码扫描脚本,在所述代码扫描脚本中添加ESLint代码检测工具;选择所述ESLint代码检测工具中的CLlEngine对象作为接口扫描对象;根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数。

[0129] 在一些实施例中,脚本运行模块403,具体用于:获取用户预先制定的扫描规则文本,将所述扫描规则文本转换成多种格式的配置文件;将多种格式的配置文件保存至所述ESLint代码检测工具的特定目录生成配置目录;从所述配置目录中读取相应的配置文件对所述接口扫描对象进行配置,得到对象属性参数。

[0130] 在一些实施例中,脚本运行模块403,具体用于:显示规则显示界面以及在所述规则显示界面中显示默认扫描规则文本;监测用户是否对所述默认扫描规则文本进行修改操作;若用户对所述默认扫描规则文本进行修改操作,则保存并获取修改后的默认扫描规则文本作为预先制定的扫描规则文本。

[0131] 调用扫描模块404,用于调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0132] 请参阅图5,图5是本申请的实施例提供另一种代码质量扫描装置的示意性框图,该代码质量扫描装置用于执行前述的代码质量扫描方法。其中,该代码质量扫描装置可以配置于服务器。

[0133] 如图5所示,该代码质量扫描装置500,包括:指令获取模块501、文件获取模块502、分类记录模块503、脚本运行模块504、调用扫描模块505和生成发送模块506。

[0134] 指令获取模块501,用于接收终端发送的用于扫描前端应用的代码的扫描指令,其

中,所述扫描指令包括存储所述前端应用对应的路径地址。

[0135] 其中,所述扫描指令为所述终端根据用户输入的前端应用的应用标识和路径地址生成的扫描指令。

[0136] 文件获取模块502,用于根据所述路径地址获取所述前端应用的各个代码文件。

[0137] 分类记录模块503,用于根据所述路径地址中路径关键字对各个代码文件进行分类,并将分类后的代码文件对应的文件名称记录在预设列表中,得到所述前端应用的文件列表。

[0138] 脚本运行模块504,用于运行预设的代码扫描脚本,其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数。

[0139] 调用扫描模块505,用于调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0140] 生成发送模块506,用于根据所述扫描结果生成扫描报告,并将所述扫描报告发送至用户。

[0141] 需要说明的是,所属领域的技术人员可以清楚地了解到,为了描述的方便和简洁,上述描述的装置和各模块的具体工作过程,可以参考前述方法实施例中的对应过程,在此不再赘述。

[0142] 上述的装置可以实现为一种计算机程序的形式,该计算机程序可以在如图6所示的计算机设备上运行。

[0143] 请参阅图6,图6是本申请的实施例提供的一种计算机设备的结构示意性框图。该计算机设备可以是服务器。

[0144] 参阅图6,该计算机设备包括通过系统总线连接的处理器、存储器和网络接口,其中,存储器可以包括非易失性存储介质和内存储器。

[0145] 非易失性存储介质可存储操作系统和计算机程序。该计算机程序包括程序指令,该程序指令被执行时,可使得处理器执行任意一种代码质量扫描方法。

[0146] 处理器用于提供计算和控制能力,支撑整个计算机设备的运行。

[0147] 内存储器为非易失性存储介质中的计算机程序的运行提供环境,该计算机程序被处理器执行时,可使得处理器执行任意一种代码质量扫描方法。

[0148] 该网络接口用于进行网络通信,如发送分配的任务等。本领域技术人员可以理解,图6中示出的结构,仅仅是与本申请方案相关的部分结构的框图,并不构成对本申请方案所应用于其上的计算机设备的限定,具体的计算机设备可以包括比图中所示更多或更少的部件,或者组合某些部件,或者具有不同的部件布置。

[0149] 应当理解的是,处理器可以是中央处理单元(Central Processing Unit,CPU),该处理器还可以是其他通用处理器、数字信号处理器(Digital Signal Processor,DSP)、专用集成电路(Application Specific Integrated Circuit,ASIC)、现场可编程门阵列(Field-Programmable Gate Array,FPGA)或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件等。其中,通用处理器可以是微处理器或者该处理器也可以是任何常规的处理器等。

[0150] 其中,所述处理器用于运行存储在存储器中的计算机程序,以实现如下步骤:

[0151] 获取用于扫描前端应用的代码的扫描指令,其中,所述扫描指令包括存储所述前端应用对应的路径地址;根据所述路径地址获取所述前端应用的文件列表;运行预设的代码扫描脚本,其中,所述代码扫描脚本中添加有用于创建接口扫描对象的代码检测工具以及根据预先制定的代码扫描规则对所述接口扫描对象进行配置生成的对象属性参数;调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果。

[0152] 在一个实施例中,所述处理器在运行存储在存储器中的计算机程序,以实现如下步骤:

[0153] 获取预先创建的代码扫描脚本,在所述代码扫描脚本中添加ESLint代码检测工具;选择所述ESLint代码检测工具中的CLIEngine对象作为接口扫描对象;根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数。

[0154] 在一个实施例中,所述处理器在实现所述根据预先制定的代码扫描规则对所述接口扫描对象进行配置,得到对象属性参数时,具体用于实现:

[0155] 获取用户预先制定的扫描规则文本,将所述扫描规则文本转换成多种格式的配置文件;将多种格式的配置文件保存至所述ESLint代码检测工具的特定目录生成配置目录;从所述配置目录中读取相应的配置文件对所述接口扫描对象进行配置,得到对象属性参数。

[0156] 在一个实施例中,所述处理器在实现所述获取用户预先制定的扫描规则文本时,具体用于实现:

[0157] 显示规则显示界面以及在所述规则显示界面中显示默认扫描规则文本;监测用户是否对所述默认扫描规则文本进行修改操作;若用户对所述默认扫描规则文本进行修改操作,则保存并获取修改后的默认扫描规则文本作为预先制定的扫描规则文本。

[0158] 在一个实施例中,所述处理器在实现所述根据前端应用的路径地址获取所述前端应用的文件列表时,具体用于实现:

[0159] 根据所述路径地址获取所述前端应用的各个代码文件;根据所述路径地址中路径关键字对各个代码文件进行分类,并将分类后的代码文件对应的文件名称记录在预设列表中,得到所述前端应用的文件列表。

[0160] 在一个实施例中,所述处理器在实现所述获取用于扫描前端应用的代码的扫描指令时,具体用于实现:

[0161] 接收终端发送的用于扫描前端应用的代码的扫描指令,其中,所述扫描指令为所述终端根据用户输入的前端应用的应用标识和路径地址生成的扫描指令。

[0162] 在一个实施例中,所述处理器在实现调用所述接口扫描对象,根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描,得到扫描结果之后,还用于实现:

[0163] 根据所述扫描结果生成扫描报告,并将所述扫描报告发送至用户。

[0164] 本申请的实施例中还提供一种计算机可读存储介质,所述计算机可读存储介质存储有计算机程序,所述计算机程序中包括程序指令,所述处理器执行所述程序指令,实现本申请实施例提供的任一项代码质量扫描方法。

[0165] 其中,所述计算机可读存储介质可以是前述实施例所述的计算机设备的内部存储单元,例如所述计算机设备的硬盘或内存。所述计算机可读存储介质也可以是所述计算机

设备的外部存储设备,例如所述计算机设备上配备的插接式硬盘,智能存储卡(Smart Media Card,SMC),安全数字(Secure Digital,SD)卡,闪存卡(Flash Card)等。

[0166] 以上所述,仅为本申请的具体实施方式,但本申请的保护范围并不局限于此,任何熟悉本技术领域的技术人员在本申请揭露的技术范围内,可轻易想到各种等效的修改或替换,这些修改或替换都应涵盖在本申请的保护范围之内。因此,本申请的保护范围应以权利要求的保护范围为准。

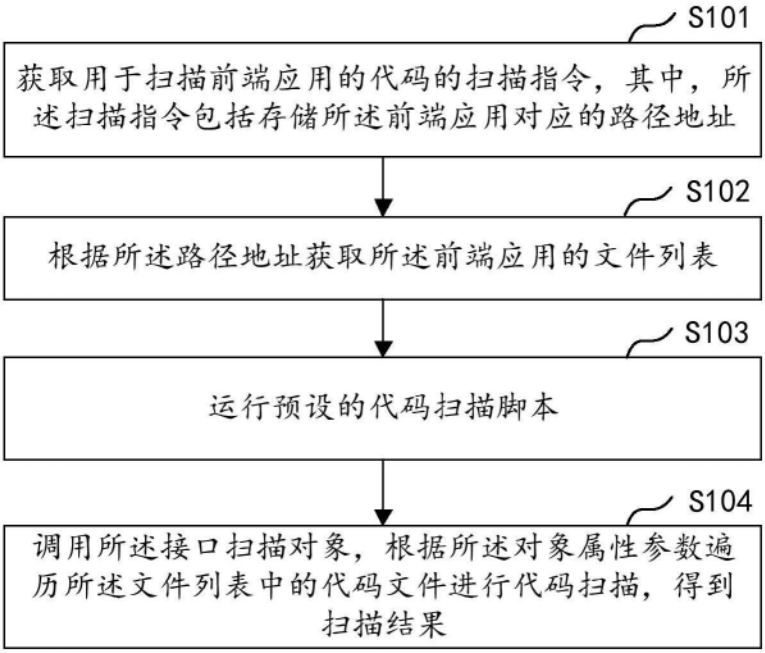

S101
获取用于扫描前端应用的代码的扫描指令，其中，所述扫描指令包括存储所述前端应用对应的路径地址
S102
根据所述路径地址获取所述前端应用的文件列表
S103
运行预设的代码扫描脚本
S104
调用所述接口扫描对象，根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描，得到扫描结果

图1

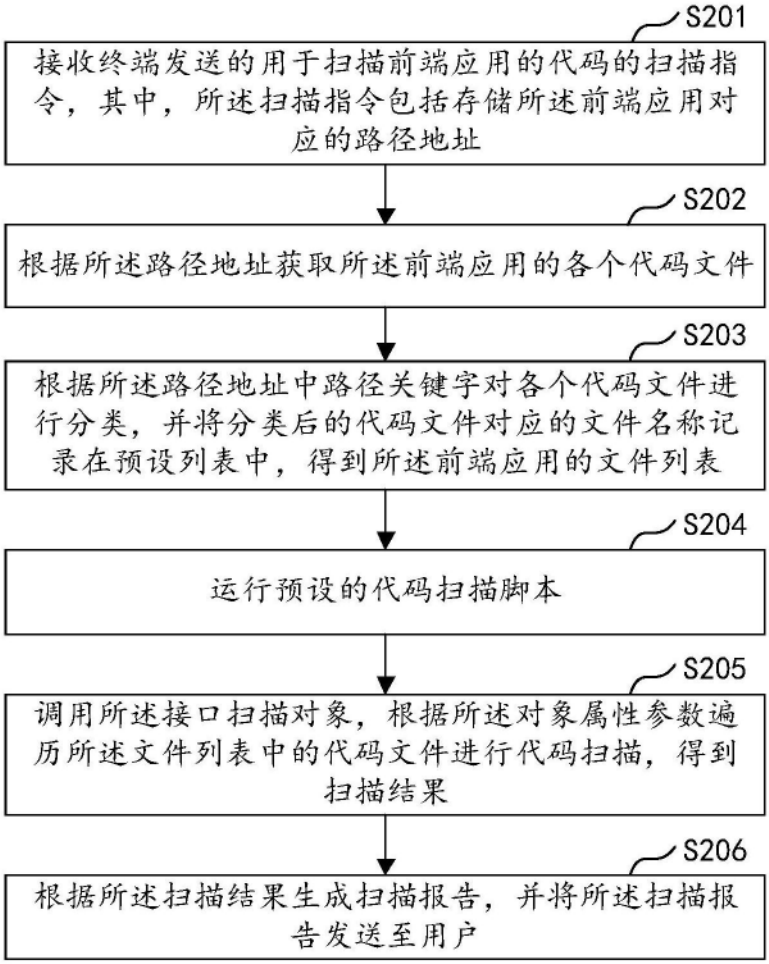

S201
接收终端发送的用于扫描前端应用的代码的扫描指令，其中，所述扫描指令包括存储所述前端应用对应的路径地址
S202
根据所述路径地址获取所述前端应用的各个代码文件
S203
根据所述路径地址中路径关键字对各个代码文件进行分类，并将分类后的代码文件对应的文件名称记录在预设列表中，得到所述前端应用的文件列表
S204
运行预设的代码扫描脚本
S205
调用所述接口扫描对象，根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描，得到扫描结果
S206
根据所述扫描结果生成扫描报告，并将所述扫描报告发送至用户

图2

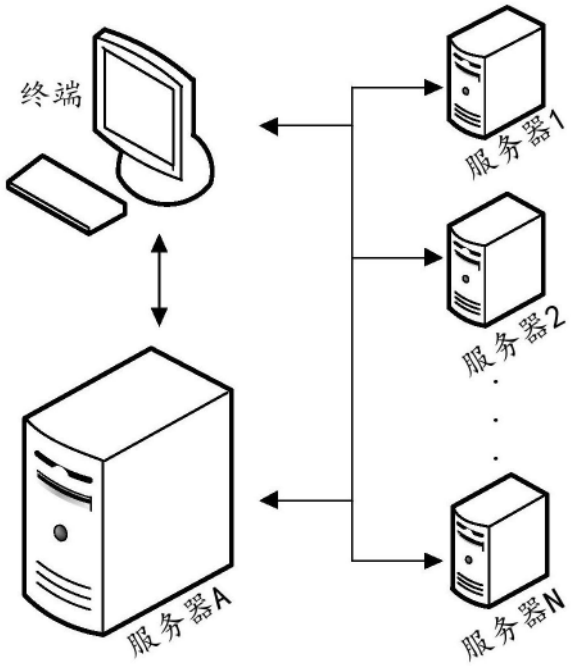


图3a

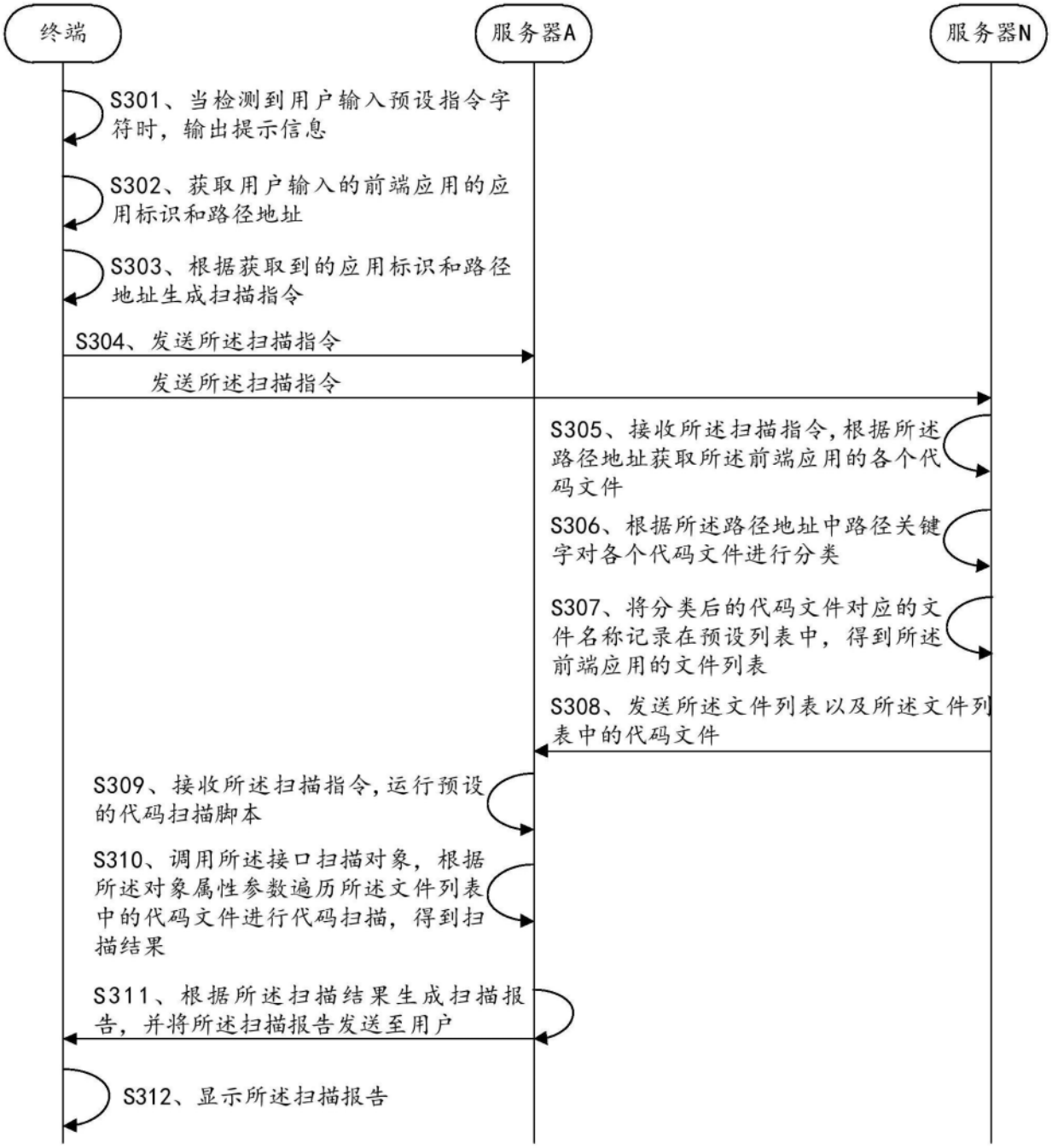

终端
服务器A
服务器N
S301、当检测到用户输入预设指令字符时，输出提示信息
S302、获取用户输入的前端应用的应用标识和路径地址
S303、根据获取到的应用标识和路径地址生成扫描指令
S304、发送所述扫描指令
发送所述扫描指令
S305、接收所述扫描指令，根据所述路径地址获取所述前端应用的各个代码文件
S306、根据所述路径地址中路径关键字对各个代码文件进行分类
S307、将分类后的代码文件对应的文件名称记录在预设列表中，得到所述前端应用的文件列表
S308、发送所述文件列表以及所述文件列表中的代码文件
S309、接收所述扫描指令，运行预设的代码扫描脚本
S310、调用所述接口扫描对象，根据所述对象属性参数遍历所述文件列表中的代码文件进行代码扫描，得到扫描结果
S311、根据所述扫描结果生成扫描报告，并将所述扫描报告发送至用户
S312、显示所述扫描报告

图3b

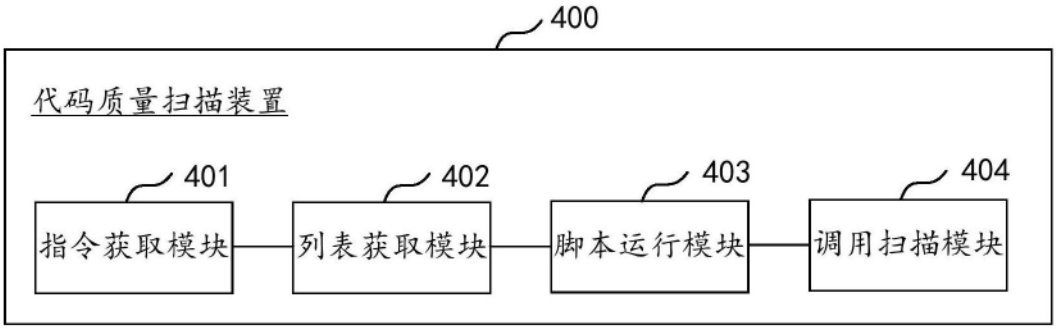

400
代码质量扫描装置
401
指令获取模块
402
列表获取模块
403
脚本运行模块
404
调用扫描模块

图4

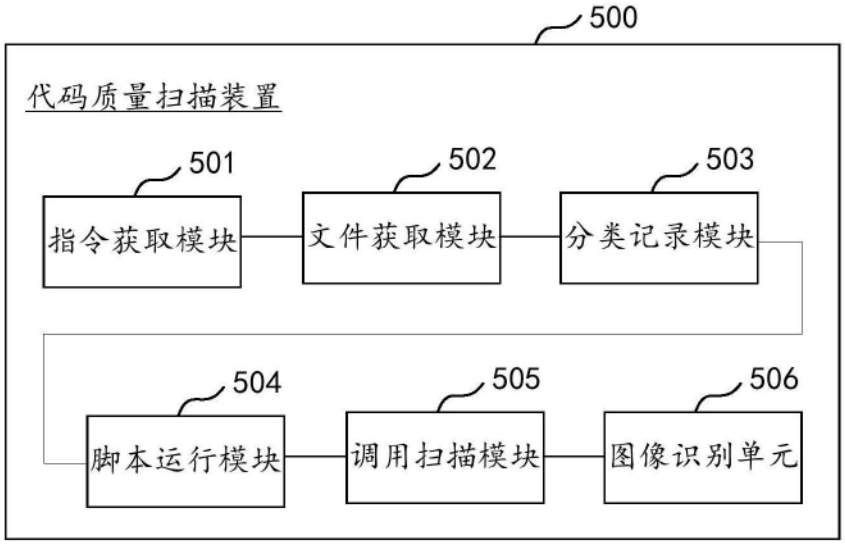

500
代码质量扫描装置
501
指令获取模块
502
文件获取模块
503
分类记录模块
504
脚本运行模块
505
调用扫描模块
506
图像识别单元

图5

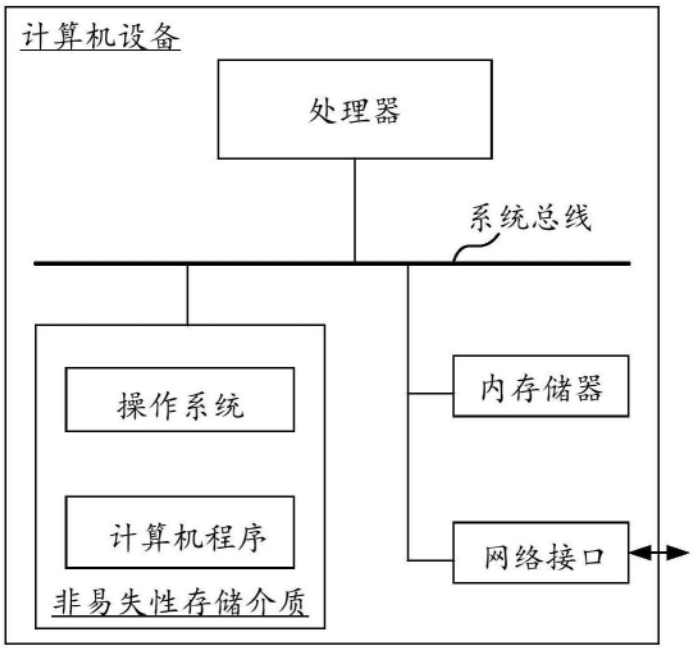


图6