(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 109032662 A
(43)申请公布日 2018.12.18

(21)申请号 201810629826.2

(22)申请日 2018.06.19

(71)申请人 上海陆家嘴国际金融资产交易市场股份有限公司
地址 200120 上海市浦东新区自由贸易试验区陆家嘴环路1333号13楼

(72)发明人 徐文灏 马奇 于乐怡

(74)专利代理机构 广州华进联合专利商标代理有限公司 44224
代理人 孙凯乐

(51)Int.Cl.
G06F 8/73(2018.01)

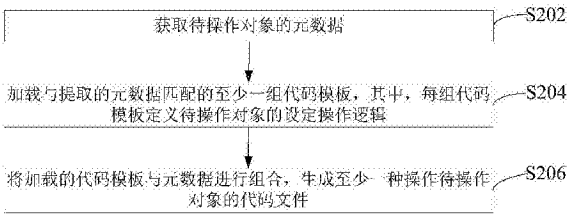
权利要求书2页 说明书11页 附图4页

(54)发明名称

代码文件生成方法、装置、计算机设备和存储介质

(57)摘要

本申请涉及一种代码文件生成方法、装置、计算机设备和存储介质。方法包括：获取待操作对象的元数据；加载与提取的所述元数据匹配的至少一组代码模板，其中，每组代码模板定义所述待操作对象的设定操作逻辑；将加载的代码模板与所述元数据进行组合，生成至少一种操作所述待操作对象的代码文件。采用本方法能够实现多种操作逻辑代码灵活定制。



CN 109032662 A

1.一种代码文件生成方法,所述方法包括:

获取待操作对象的元数据;

加载与提取的所述元数据匹配的至少一组代码模板,其中,每组代码模板定义所述待操作对象的设定操作逻辑;

将加载的代码模板与所述元数据进行组合,生成至少一种操作所述待操作对象的代码文件。

2.根据权利要求1所述的方法,其特征在于,所述获取所述操作对象的元数据,包括:

获取待操作对象的源码文件;

解析所述源码文件,根据预先配置的源码文件类型与数据提取策略之间的对应关系确定所述源码文件对应的数据提取策略;

加载所述数据提取策略,通过所述数据提取策略从所述源码文件中提取元数据。

3.根据权利要求1所述的方法,其特征在于,所述加载与提取的所述元数据匹配的至少一组代码模板,包括:

根据所述元数据的类型调用至少一个模板处理器,其中,所述模板处理器为设定代码模板组合的加载器;

通过所述模板处理器加载对应的代码模板组合。

4.根据权利要求1-3任一项所述的方法,其特征在于,所述将加载的代码模板与所述元数据进行组合,生成至少一种操作所述待操作对象的代码文件,包括:

将提取的所述元数据转换成键值对的集合,所述键值对包括变量和变量值;

获取所述代码模板目录,提取所述代码模板目录中的目录子项,将所述目录子项中的变量替换成所述元数据中对应变量的变量值,根据变量替换后的所述目录子项生成包名;

根据所述代码模板目录中的文件子项获取代码模板文件,将所述代码模板文件中的变量替换成所述元数据中对应变量的变量值,生成代码文件,其中,生成的所述代码文件声明生成的所述包名。

5.根据权利要1-3任一项所述的方法,其特征在于,在将加载的代码模板与所述元数据进行组合,生成至少一种操作所述待操作对象的代码文件后,还包括:

判断生成的所述代码文件是否已经存在,若是,则将生成的所述代码文件和已有代码文件转换成语法树,循环执行配置的合并策略生成合并后的语法树,根据合并后的所述语法树生成合并后的代码文件。

6.一种代码文件生成装置,其特征在于,所述装置包括:

元数据提取模块,用于获取所述待操作对象的元数据;

代码模板加载模块,用于加载与提取的所述元数据匹配的至少一组代码模板,其中,每组所述代码模板定义所述待操作对象的设定操作逻辑;

组合模块,用于将加载的代码模板与所述元数据进行组合,生成至少一种操作所述待操作对象的代码文件。

7.根据权利要求6所述的装置,其特征在于,所述代码模板加载模块,还用于根据所述元数据的类型调用至少一个预先定义的模板处理器;通过所述模板处理器加载代码模板组合,预先为每个所述模板处理器配置其可加载的代码模板组合。

8.根据权利要求6或7所述的装置,其特征在于,所述组合模块,还包括:

格式转换模块，用于将提取的所述元数据转换成键值对的集合，所述键值对包括变量和变量值；

包名生成模块，用于获取所述代码模板目录，提取所述代码模板目录中的目录子项，将所述目录子项中的变量替换成所述元数据中对应变量的变量值，根据变量替换后的所述目录子项生成包名；

代码文件生成模块，用于根据所述代码模板目录中的文件子项获取代码模板文件，将所述代码模板文件中的变量替换成所述元数据中对应变量的变量值，生成代码文件，其中，生成的所述代码文件声明生成的所述包名。

9.一种计算机设备，包括存储器和处理器，所述存储器存储有计算机程序，其特征在于，所述处理器执行所述计算机程序时实现权利要求1至5中任一项所述方法的步骤。

10.一种计算机可读存储介质，其上存储有计算机程序，其特征在于，所述计算机程序被处理器执行时实现权利要求1至5中任一项所述的方法的步骤。

代码文件生成方法、装置、计算机设备和存储介质

技术领域

[0001] 本申请涉及计算机技术领域，特别是涉及一种代码文件生成方法、装置、计算机设备和存储介质。

背景技术

[0002] 随着互联网技术的发展，当下无疑已经进入了全民互联网时代，而软件系统作为互联网的灵魂，为满足人们日益增长的计算机软件服务需求，软件开发人员需要不断进行计算机程序的开发和创新。

[0003] 计算机程序由一系列的代码组合而成且能够表达特定数学逻辑。生成上述的一系列的代码的传统方式包括：开发人员手动敲写代码和通过代码生成工具自动生成。其中，开发人员纯手动敲写代码的方式的缺点在于大量代码重复编写，费时费力。而通过工具生成代码的方式，由于传统的代码生成工具多为针对某个特定场景进行代码生成，一种代码生成工具只能生成一种特定场景(如数据库操作)的代码。生成多种场景下的操作代码时需要使用多种代码生成工具，代码生成效率低，灵活性差。

发明内容

[0004] 基于此，有必要针对上述技术问题，提供一种能够实现多种操作逻辑代码灵活定制，代码生成效率更高的代码生成文件方法、装置、计算机设备和存储介质。

[0005] 一种代码文件生成方法，所述方法包括：

[0006] 获取待操作对象的元数据；

[0007] 加载与提取的所述元数据匹配的至少一组代码模板，其中，每组代码模板定义所述待操作对象的设定操作逻辑；

[0008] 将加载的代码模板与所述元数据进行组合，生成至少一种操作所述待操作对象的代码文件。

[0009] 在一个实施例中，所述获取所述操作对象的元数据，包括：

[0010] 获取待操作对象的源码文件；

[0011] 解析所述源码文件，根据预先配置的源码文件类型与数据提取策略之间的对应关系确定所述源码文件对应的数据提取策略；

[0012] 加载所述数据提取策略，通过所述数据提取策略从所述源码文件中提取元数据。

[0013] 在一个实施例中，所述加载与提取的所述元数据匹配的至少一组代码模板，包括：

[0014] 根据所述元数据的类型调用至少一个模板处理器，其中，所述模板处理器为设定代码模板组合的加载器；

[0015] 通过所述模板处理器加载对应的代码模板组合。

[0016] 在一个实施例中，所述将加载的代码模板与所述元数据进行组合，生成至少一种操作所述待操作对象的代码文件，包括：

[0017] 将提取的所述元数据转换成键值对的集合，所述键值对包括变量和变量值；

[0018] 获取所述代码模板目录，提取所述代码模板目录中的目录子项，将所述目录子项中的变量替换成所述元数据中对应变量的变量值，根据变量替换后的所述目录子项生成包名；

[0019] 根据所述代码模板目录中的文件子项获取代码模板文件，将所述代码模板文件中的变量替换成所述元数据中对应变量的变量值，生成代码文件，其中，生成的所述代码文件声明生成的所述包名。

[0020] 在一个实施例中，在将加载的代码模板与所述元数据进行组合，生成至少一种操作所述待操作对象的代码文件后，还包括：

[0021] 判断生成的所述代码文件是否已经存在，若是，则将生成的所述代码文件和已有代码文件转换成语法树，循环执行配置的合并策略生成合并后的语法树，根据合并后的所述语法树生成合并后的代码文件。

[0022] 一种代码文件生成装置，所述装置包括：

[0023] 元数据提取模块，用于获取所述待操作对象的元数据；

[0024] 代码模板加载模块，用于加载与提取的所述元数据匹配的至少一组代码模板，其中，每组所述代码模板定义所述待操作对象的设定操作逻辑；

[0025] 组合模块，用于将加载的代码模板与所述元数据进行组合，生成至少一种操作所述待操作对象的代码文件。

[0026] 在一个实施例中，所述代码模板加载模块，还用于根据所述元数据的类型调用至少一个预先定义的模板处理器；通过所述模板处理器加载代码模板组合，预先为每个所述模板处理器配置其可加载的代码模板组合。

[0027] 在一个实施例中，所述组合模块，还包括：

[0028] 格式转换模块，用于将提取的所述元数据转换成键值对的集合，所述键值对包括变量和变量值；

[0029] 包名生成模块，用于获取所述代码模板目录，提取所述代码模板目录中的目录子项，将所述目录子项中的变量替换成所述元数据中对应变量的变量值，根据变量替换后的所述目录子项生成包名；

[0030] 代码文件生成模块，用于根据所述代码模板目录中的文件子项获取代码模板文件，将所述代码模板文件中的变量替换成所述元数据中对应变量的变量值，生成代码文件，其中，生成的所述代码文件声明生成的所述包名。

[0031] 一种计算机设备，包括存储器和处理器，所述存储器存储有计算机程序，所述处理器执行所述计算机程序时实现权利要求1至5中任一项所述方法的步骤。

[0032] 一种计算机可读存储介质，其上存储有计算机程序，所述计算机程序被处理器执行时实现权利要求1至5中任一项所述的方法的步骤。

[0033] 上述代码生成方法、装置、计算机设备和存储介质，通过扫描待操作对象，提取能够描述对象属性的元数据，加载能够与元数据组合的一组或者多组代码模板，每组代码模板可实现对待操作对象的设定场景中特定类型的操作，组合元数据和代码模板生成代码文件，生成的代码文件包括对待操作对象实施的多种操作的代码文件，通过灵活加载不同的代码模板，实现了对待操作对象的操作代码的灵活定制，一次操作即可生成多种场景下的多种操作逻辑的代码文件。

附图说明

[0034] 图1为一个实施例中代码文件生成方法的应用场景图；

[0035] 图2为一个实施例中代码文件生成方法的流程示意图；

[0036] 图3为一个实施例中提取元数据的原理图；

[0037] 图4为一个实施例中加载与元数据匹配的至少一组代码模板所涉及的流程示意图；

[0038] 图5为一个实施例中组合元数据和代码模板，生成操作对象的代码文件所涉及的流程示意图；

[0039] 图6为一个实施例中代码文件合并的原理图；

[0040] 图7为一个实施例中代码文件生成装置的结构框图；

[0041] 图8为一个实施例中组合模块的结构框图；

[0042] 图9为另一个实施例中代码文件生成装置的结构框图；

[0043] 图10为一个实施例中计算机设备的内部结构图。

具体实施方式

[0044] 为了使本申请的目的、技术方案及优点更加清楚明白，以下结合附图及实施例，对本申请进行进一步详细说明。应当理解，此处描述的具体实施例仅仅用以解释本申请，并不用于限定本申请。

[0045] 本申请提供的代码生成方法，可以应用于如图1所示的应用环境中。其中，终端102通过网络与服务器104通过网络进行通信。终端102用于提供用户操作界面，监听用户指定的待操作对象、监听用户的配置操作等，向服务器104发送代码生成请求等。服务器104提取待操作对象的元数据，加载与元数据匹配的至少一组代码模板，将元数据和代码模板进行组合，生成对待操作对象的至少一种操作类型的代码。终端102可以但不限于是各种个人计算机、笔记本电脑、智能手机、平板电脑，服务器104可以用独立的服务器或者是多个服务器组成的服务器集群来实现。

[0046] 在一个实施例中，如图2所示，提供了一种代码文件生成方法，以该方法应用于图1中的服务器为例进行说明，包括以下步骤：

[0047] 步骤S202：获取待操作对象的元数据。

[0048] 待操作对象可以是一个web页面、接口或者类等具有相对固化属性、且能被进一步操作的程序单元。待操作对象的元数据是指用来描述对象属性的数据。例如，一个http接口的元数据可以包括接口路径、接口的输入输出参数等。web页面的元数据可以是组件id，组件类型等。

[0049] 扫描待操作对象的源代码，从待操作对象的源代码中提取元数据。例如JAVA工程，通过加载待操作对象的类文件，遍历类文件中的类方法(method)和类变量(field)获取类文件中描述类属性元数据。

[0050] 步骤S204：加载与提取的元数据匹配的至少一组代码模板，其中，每组代码模板定义待操作对象的设定操作逻辑。

[0051] 抽象针对各种类型的待操作对象可能实施的各种类型的操作逻辑，定义用于操作

各种类型对象的代码模板。操作逻辑是抽象出来的操作方法框架，一种类型的操作逻辑实质上是能够对同一类待操作对象实施设定操作的代码。如对“http接口”实施“调用”操作的操作代码为“接口调用代码模板”。

[0052] 针对一种类型的待操作对象可能实施的操作有多个，本实施例中，针将对于该待操作对象定义多个代码模板，每个代码模板实现对待操作对象的一种操作逻辑。如待操作对象为接口，针对“http接口”定义的代码模板可能是调用接口的代码模板、封装接口的代码模板等等。针对于同一对象的多个代码模板的组合也可以用于实现一种操作逻辑。

[0053] 在一个实施例中，可以在定义的每个代码模板中添加匹配对象的标签，加载的代码模板为带有待操作对象标签的代码模板。在另一个实施例中，可以计算提取的元数据与代码模板中可替换内容的匹配度，根据匹配度确定代码模板是否适应对该待操作对象实施相应的操作。还可以通人工配置的方式，用户选定当前需要加载的代码模板。

[0054] 本实施例中，加载的与元数据匹配的组代码模板可以包括一个代码模板，也可以包括多个代码模板，无论是一个代码模板还是多个代码模板的组合都能够实现对待操作对象的一种操作逻辑。加载一组代码模板即加载一组能够实现一种操作逻辑的代码模板。本实施例侧重加载多组代码模板，多组代码模板可实现对待操作对象的多种类型的操作逻辑。

[0055] 步骤S206：将加载的代码模板与元数据进行组合，生成至少一种操作待操作对象的代码文件。

[0056] 模板引擎使用待操作对象的元数据替换加载的每一组代码模板的模板内容，替换后的每一组代码模板可实现对待操作对象的设定类型的操作逻辑。

[0057] 本实施例中，通过扫描待操作的对象，提取能够描述对象属性的元数据，加载能够与元数据组合的一组或者多组代码模板，每组代码模板可实现对待操作对象的特定类型的操作，组合元数据和代码模板，即可生成针对待操作对象的各种类型操作逻辑的代码。本实施例可批量生成待操作对象的多种操作类型的代码，生成的代码可根据需要灵活定制。与待操作对象匹配的代码模板可进行定制化的组合，以衍生出多个对待操作对象的操作逻辑。

[0058] 此外，通过针对某一类型的待操作对象抽象代码逻辑，减少了开发过程中代码的大量重复敲写。

[0059] 在一个实施例中，步骤S202：获取待操作对象元数据包括：获取待操作对象的源码文件；解析源码文件，根据预先配置的源码文件类型与数据提取策略之间的对应关系确定源码文件对应的数据提取策略；加载确定的数据提取策略，通过数据提取策略从待操作对象的源码文件中提取元数据。

[0060] 进一步的，请求中可携带多个待操作对象的标识，服务器采用多线程的方式同时针对多个待操作对象加载相应的提取策略，同时提取多个待操作对象的元数据。

[0061] 如图3所示，服务器加载两种类型的待操作对象的源码文件，如java.class文件(类文件)和Html文件，加载预先为java.class文件配置path策略，Method策略，param策略，加载预先为Html文件配置的组件策略，通过加载的策略提取待操作对象的元数据并输出，输出的元数据用于与匹配的代码模板进行匹配。

[0062] 本实施例中，针对不同类型的待操作对象定义不同的提取策略，提取待操作对象

的元数据时直接调用提取策略即可实现对元数据的自动提取。不同类型的待操作对象的元数据可通过调用不通的提取策略同时提取，提高了数据提取效率。

[0063] 在一个实施例中，如图4所示，步骤S204：加载与元数据匹配的至少一组代码模板，包括如下步骤：

[0064] 步骤S402：根据元数据的类型调用至少一个模板处理器，其中，模板处理器为设定代码模板组合的加载器。

[0065] 步骤S404：通过模板处理器加载对应的代码模板组合。

[0066] 预先定义多个模板处理器，具体定义每一个模板处理器能够加载的代码模板组合。即为每个模板处理器配置可加载的代码模板组合，模板处理器实质上为设定代码模板组合的加载器。通过模板处理器可加载的代码模板组合，以及代码模板组合中的代码模板所适应的元数据类型，建立模板处理器与元数据类型之间的关联关系。具体实现时表现为如下步骤：根据元数据的类型调用至少一个模板处理器，通过模板处理器加载其能够加载的代码模板组合。同一模板处理器加载的多个代码模板可以实现对待操作对象的一种操作逻辑，也可以实现对待操作对象的一系列相关的操作逻辑。

[0067] 本实施例通过定义模板处理器，可预先对各种待处理对象所匹配的所有的代码模板进行组合分类，服务器只需根据元数据的类型查找相匹配的模板处理器，即能够按照预先定义好的组合去加载代码模板，无需在进行繁琐的代码模板的组合。

[0068] 举例来说，与某一类型的待操作对象匹配的代码模板包括代码模板A，代码模板B和代码模板C，其中，代码模板B和代码模板C的组合，以及代码A、代码模板B和代码模板C的组合也能实现对待操作对象的一种类型的操作，那么，可定义5个模板处理器，定义这5个模板处理器可加载的代码模板依次为代码模板A、代码模板B、代码模板C、代码模板B和代码模板C的组合、代码A、代码模板B和代码模板C的组合。当然，如果还有其他组合也可以实现对待操作对象的操作，则定义更多的模板处理器。为定义的模板处理器添加对应的元数据类型标签，其中一个模板处理器可对应多个元数据类型标签。

[0069] 在一个实施例中，如图5所示，步骤S306：将加载的代码模板与元数据进行组合，生成至少一种操作待操作对象的代码文件，包括如下步骤：

[0070] 需要说明的是，下述步骤给出的是元数据与一个或者一组代码模板组合的逻辑，可通过循环执行下述步骤实现元数据与多组代码模板的组合，或者多线程运行下述步骤，实现元数据与多组代码模板的同步组合。

[0071] 步骤S502：将元数据转换成键值对的集合，键值对包括变量和变量值。

[0072] 将元数据转换成("key＝value")字符串，key为变量，value为变量值。

[0073] 步骤S504：获取代码模板的目录，将目录中目录子项的变量替换成元数据中的变量值。

[0074] 递归遍历代码模板目录中的所有子项，判断当前遍历的子项是文件还有目录，若是目录，查找目录中是否有变量，若是，则查找元数据中是否存在对应目录中的变量的变量值，若是，用对应的变量值替换目录中的变量，直至替换掉代码模板每层目录子项的变量。

[0075] 如模板目录中带有${var}变量字符，转换成的元数据为context.put("var","hello")，目录遍历时，将${var}字符用"hello"替换。

[0076] 步骤S506：根据替换变量后的代码模板目录的目录子项生成包名。

[0077] 步骤S508:获取代码模板文件,使用元数据替换文件中的内容,生成代码文件,生成的代码文件声明生成的包名。

[0078] 代码模板目录包括目录子项和文件子项,将目录子项中的变量替换成元数据的变量值后,基于代码模板的各层目录子项按照设定规则生成包名。

[0079] 代码模板目录中的文件子项指定代码模板文件,模板引擎将代码模板文件中的内容用元数据进行替换,去掉模板中的空行和注解,生成代码文件。

[0080] 本实施例中,根据模板目录的目录子项生成包名,根据模板目录中的文件子项生成代码文件,代码文件声明生成的包名。在生成代码的时候,为生成的代码文件声明一个包名,即使用该包名为代码文件中的类创建新的命名空间,由于包创建了新的命名空间(namespace),所以不会跟其他包中的任何名字产生命名冲突。使用包机制,更容易实现访问控制,并且让定位相关类更加简单,不使用这种包机制为新生成得到代码声明包名将会出现编辑错误。

[0081] 在一个实施例中,在提取待操作对象的元数据时,为元数据配置包名前缀。可以是,查找待操作对象所处系统的系统标识名,将所处系统的系统标识名配置为待操作对象的元数据的包名前缀。生成的包名中添加包名前缀。

[0082] 举例来说,代码模板的模板目录为:templates/app/AppCaller.java,其中/app/为目录子项,AppCaller.java为文件子项,若配置的包名前缀为com.lufax,根据目录子项生成的包名为com.lufax.app。基于文件子项加载模板文件,组合模板文件和元数据后生成代码文件,该代码文件对应的包名即为com.lufax.app。

[0083] 当加载的一组代码模板中包括多个代码模板时,按照步骤S502-S508生成多个代码文件,每个代码文件对应生成一个包名。生成得到代码文件的目录包括其对应的包名。

[0084] 应该理解的是,虽然图2、4-5的流程图中的各个步骤按照箭头的指示依次显示,但是这些步骤并不是必然按照箭头指示的顺序依次执行。除非本文中有明确的说明,这些步骤的执行并没有严格的顺序限制,这些步骤可以以其它的顺序执行。而且,图2、4-5中的至少一部分步骤可以包括多个子步骤或者多个阶段,这些子步骤或者阶段并不必然是在同一时刻执行完成,而是可以在不同的时刻执行,这些子步骤或者阶段的执行顺序也不必然是依次进行,而是可以与其它步骤或者其它步骤的子步骤或者阶段的至少一部分轮流或者交替地执行。

[0085] 在一个实施例中,在步骤S508:获取述所代码模板的文件,使用元数据替换文件中的内容生成代码文件,其中,生成的代码文件声明根据代码模板目录生成的包名之后,还包括:判断生成的代码文件是否已经存在,若是,则合并代码文件。

[0086] 如图6为代码合并的逻辑示意图。首先获取已有代码文件和新代码文件,使用Eclipse JDT库将已有代码和新代码转换成语法树,加载配置的合并策略并循环执行合并策略以生成合并后的语法树,将合并后的语法树转换成代码文本,将代码文本写入目标代码文件。

[0087] 在一个实施例中,还包括目录和目录的合并。具体为:判断合并源的类型,是目录和目录的合并还是文件和文件的合并。若是文件和文件的合并,采用如图6所示的合并方法进行合并。若是目录和目录的合并,则,读取源目录中的文件列表,遍历其中的每一个文件,获得文件与源目录路径之间的相对路径,根据相对路径,获取目标文件的完整路径,判断目

标文件是否存在，若不存在，将当前遍历的代码文件中的代码复制到目标代码文件中，若目标文件存在，则进行按照图6所示的方法进行源文件和目标文件的合并。遍历源目录中的下一个源文件，直至遍历完源目录文件列表中的所有的文件。

[0088] 本实施例中，预先定义如下表格所示合并策略，表1中给出的合并策略类型、名称和对合并策略的描述，具体参考如下的表格。

[0089] 表1

[0090]

类型	名称	简介
Java	CoreMergeTypeStrategy	合并代码的核心策略类，负责合并2个类型，触发合并策略，并递归合并所有内部类型。
	PathRegexStrategyDispatcher	根据source文件的路径来进行分发的策略
	AddNewImportStrategy	该策略用于向目标代码添加import语句
	AddNewEntireTypeStrategy	该策略用于当目标代码中没有定义类型时，添加source中的整个类型到目标代码
Type	AddNewAnnotationStrategy	该策略用于向目标代码添加新的Annotation。
	AddNewConstructorStrategy	该策略用于根据source中的构造函数列表，添加新增的构造函数到目标代码。
	AddNewEntireInnerTypeStrategy	该策略用于向目标代码中添加新增的内部类定义。
	AddNewFieldStrategy	该策略用于根据source中的非static field列表，添加新增的非static field到目标代码。
	AddNewMethodStrategy	该策略用于根据source中的普通非static函数列表，添加新增的函数到目标代码。
	AddNewStaticFieldStrategy	该策略用于根据source中的static field列表，添加新增的static field到目标代码。
	AddNewStaticMethodStrategy	该策略用于根据source中的static函数列表，添加新增的static函数到目标代码。
	AddNewSuperClassStrategy	该策略用于根据source中的基类，添加基类设置。
	AddNewSuperInterfaceStrategy	该策略用于根据source中的实现接口列表，添加新增的接口到目标代码。
	MergeTypeStrategyDispatcher	合并Type的Dispatcher，根据条件将数据流转到不同的IMergeTypeStrategy处理.
	UpdateEntireConstructorStrategy	该策略用于合并source和target中所有共有的构造函数，将source中的新构造函数，覆盖掉原来的构造函数。
	UpdateEntireFieldStrategy	该策略用于合并source和target中所有共有的field，将source中的新field，覆盖掉原来的field。
	UpdateEntireMethodStrategy	该策略用于合并source和target中所有共有的方法，将source中的新方法，覆盖掉原来的方法。

[0091]

	UpdateMetaTypeStrategy	该策略用于根据 source 中的 type 类型，即是接口还是非接口，来更新目标代码。
Field	UpdateMergeFieldAccessibleStrategy	该策略用于合并 field 的 public, private 修饰符
	UpdateMergeFieldAssignmentStrategy	该策略用于合并 field 的赋值表达式
	UpdateMergeFieldStaticStrategy	该策略用于合并 field 的 static 修饰符
Method	MergeStatementsStrategy	合并 Method 中各条语句的策略. 默认只合并判断为 INSERT 类型的语句, CHANGE 和 DELETE 会被忽略.
	UpdateMergeMethodAccessibleStrategy	该策略用于合并 method 的 public, private 修饰符
	UpdateMergeMethodStaticStrategy	该策略用于合并 method 的 static 修饰符
	UpdateMethodReturnTypeStrategy	该策略用于更新方法的返回值类型

[0092] 本实施例中，为避免生成的代码文件覆盖已有代码，造成代码文件丢失，本实施例中预先定义代码合并策略，自动查找待合并的代码文件，包括文件和文件的合并以及目录和目录的合并。本实施例中，通过循环执行代码策略自动进行代码的合并，无需手动合并，在批量生成代码的基础上实现了代码的自动管理。

[0093] 在一个实施例中提供一种代码文件生成方法，包括如下内容：

[0094] 获取待操作对象的源码文件，从源码文件中提取待操作对象的元数据。将所有待操作对象的元数据以元数据文件的形式进行存储。获取到批量生成代码的指令时，递归遍历每一个元数据文件，从文件中读取元数据，将读取的元数据转换成递归的键值对格式的Map。创建数据容器，使用从Spring(Spring是一个分层的JavaSE/EE full-stack一站式轻量级开源框架)中加载配置的各种工具类将格式转换后的元数据加入到数据容器中，获取配置的包名前缀，将报名前缀加入到数据容器中。

[0095] 针对当前遍历的元数据配置需要加载的模板处理器，也可以根据预先定义的模板处理器与待操作对象的对应关系，自动加载对应的模板处理器。预先定义每个模板处理器加载的代码模板。遍历模板处理器能够加载的代码模板目录的所有子项，判断当前遍历的子项是文件还是目录，若是目录，查找中是否存在变量，将变量对数据容器中的键值对进行匹配，将变量替换成变量值，替换后结合当前层目录和上一层目录生成包名，遍历下一层目录子项，不断更新包名，直至遍历的子项为文件，将最新的包名加入至数据容器中，包名和包名前缀组合成完整的包名。模板引擎加载模板代码文件，调用数据容器，首先是调用数据容器中的元数据，将代码模板中的内容替换成元数据，其次，是将数据容器中的包名写入代码模板中声明包名的位置，最后替换模板内容，去除模板造成的空行以及模板中的注解，生成并输出代码文件，生成的代码文件声明数据容器中的包名，判断代码文件是否存在，若是，合并代码文件。

[0096] 遍历模板处理器可加载的下一个代码模板目录，遍历目录子项生成包名，将当前生成的包名加入至数据容器中替换已有的包名，加载当前代码模板目录对应的代码模板文件，生成代码文件，生成的代码文件声明数据容器中替换后的包名。直至遍历完该元数据对应的所有的模板处理器中的所有代码模板。获取下一个元数据文件，基于下一个元数据文件重新构建数据容器，遍历完所有的元数据文件。

[0097] 本实施例中，基于一个待操作对象生成一个数据容器，配置的众多代码模板调用该数据容易进行内容替换即可批量生成代码文件，代码文件生成效率更高。

[0098] 在一个实施例中，如图7所示，提供了一种代码文件生成装置，包括：元数据提取模块702、代码模板加载模块704和组合模块706，其中：

[0099] 元数据提取模块702，获取待操作对象的元数据。

[0100] 代码模板加载模块704，用于加载与提取的元数据匹配的至少一组代码模板，其中，每组代码模板定义待操作对象的设定操作逻辑。

[0101] 组合模块706，用于将加载的代码模板与元数据进行组合，生成至少一种操作待操作对象的代码文件。

[0102] 在一个实施例中，元数据提取模块702，还用于获取待操作对象的源码文件；解析源码文件，根据预先配置的源码文件类型与数据提取策略之间的对应关系确定源码文件对应的数据提取策略；加载确定的数据提取策略，通过数据提取策略从源码文件中提取元数据。

[0103] 在一个实施例中，代码模板加载模块704，还用于根据元数据的类型调用至少一个模板处理器，其中，模板处理器为设定代码模板组合的加载器；通过模板处理器加载对应的代码模板组合。

[0104] 在一个实施例中，如图8所示，组合模块706，包括格式转换模块802，包名生成模块804和代码文件生成模块806，其中：

[0105] 格式转换模块802，用于将元数据转换成键值对的集合，键值对包括变量和变量值。

[0106] 包名生成模块804，用于获取代码模板目录，提取代码模板目录中的目录子项，将目录子项中的变量替换成元数据中对应变量的变量值，根据变量替换后的目录子项生成包名。

[0107] 代码文件生成模块806，用于根据代码模板目录中的文件子项获取代码模板文件，将代码模板文件中的变量替换成元数据中对应变量的变量值，生成代码文件，其中，生成的代码文件声明生成的包名。

[0108] 在一个实施例中，如图9所示，代码生成装置还包括文件合并模块708，用于判断生成的代码文件是否已经存在，若是，则将生成的代码文件和已有代码文件转换成语法树，循环执行配置的合并策略生成合并后的语法树，根据合并后的语法树生成合并后的代码文件。

[0109] 关于代码生成装置的具体限定可以参见上文中对于代码生成方法的限定，在此不再赘述。上述代码生成装置中的各个模块可全部或部分通过软件、硬件及其组合来实现。上述各模块可以硬件形式内嵌于或独立于计算机设备中的处理器中，也可以以软件形式存储于计算机设备中的存储器中，以便于处理器调用执行以上各个模块对应的操作。

[0110] 在一个实施例中，提供了一种计算机设备，该计算机设备可以是服务器，其内部结构图可以如图10所示。该计算机设备包括通过系统总线连接的处理器、存储器、网络接口和数据库。其中，该计算机设备的处理器用于提供计算和控制能力。该计算机设备的存储器包括非易失性存储介质、内存储器。该非易失性存储介质存储有操作系统、计算机程序和数据库。该内存储器为非易失性存储介质中的操作系统和计算机程序的运行提供环境。该计算

机设备的数据库用于存储生成的代码文件。该计算机设备的网络接口用于与外部的终端通过网络连接通信。该计算机程序被处理器执行时以实现一种代码文件生成方法。

[0111] 本领域技术人员可以理解，图10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。

[0112] 在一个实施例中，提供了一种计算机设备，包括存储器和处理器，该存储器存储有计算机程序，该处理器执行计算机程序时实现以下步骤：获取待操作对象的元数据；加载与提取的元数据匹配的至少一组代码模板，其中，每组代码模板定义待操作对象的设定操作逻辑；将加载的代码模板与元数据进行组合，生成至少一种操作待操作对象的代码文件。

[0113] 在一个实施例中，处理器执行计算机程序时还实现以下步骤：获取待操作对象的源码文件；解析源码文件，根据预先配置的源码文件类型与数据提取策略之间的对应关系确定源码文件对应的数据提取策略；加载数据提取策略，通过数据提取策略从源码文件中提取元数据。

[0114] 在一个实施例中，处理器执行计算机程序时还实现以下步骤：根据元数据的类型调用至少一个模板处理器，其中，模板处理器为设定代码模板组合的加载器；通过模板处理器加载对应的代码模板组合。

[0115] 在一个实施例中，处理器执行计算机程序时还实现以下步骤：将提取的元数据转换成键值对的集合，键值对包括变量和变量值；获取代码模板目录，提取代码模板目录中的目录子项，将目录子项中的变量替换成元数据中对应变量的变量值，根据变量替换后的目录子项生成包名；根据代码模板目录中的文件子项获取代码模板文件，将代码模板文件中的变量替换成元数据中对应变量的变量值，生成代码文件，其中，生成的代码文件声明生成的包名。

[0116] 在一个实施例中，处理器执行计算机程序时还实现以下步骤：判断生成的代码文件是否已经存在，若是，则将生成的代码文件和已有代码文件转换成语法树，循环执行配置的合并策略生成合并后的语法树，根据合并后的语法树生成合并后的代码文件。

[0117] 在一个实施例中，提供了一种计算机可读存储介质，其上存储有计算机程序，计算机程序被处理器执行时实现以下步骤：获取待操作对象的元数据；加载与提取的元数据匹配的至少一组代码模板，其中，每组代码模板定义待操作对象的设定操作逻辑；将加载的代码模板与元数据进行组合，生成至少一种操作待操作对象的代码文件。

[0118] 在一个实施例中，计算机程序被处理器执行时还实现以下步骤：获取待操作对象的源码文件；解析源码文件，根据预先配置的源码文件类型与数据提取策略之间的对应关系确定源码文件对应的数据提取策略；加载数据提取策略，通过数据提取策略从源码文件中提取元数据。

[0119] 在一个实施例中，计算机程序被处理器执行时还实现以下步骤：根据元数据的类型调用至少一个模板处理器，其中，模板处理器为设定代码模板组合的加载器；通过模板处理器加载对应的代码模板组合。

[0120] 在一个实施例中，计算机程序被处理器执行时还实现以下步骤：将提取的元数据转换成键值对的集合，键值对包括变量和变量值；获取代码模板目录，提取代码模板目录中的目录子项，将目录子项中的变量替换成元数据中对应变量的变量值，根据变量替换后的

目录子项生成包名;根据代码模板目录中的文件子项获取代码模板文件,将代码模板文件中的变量替换成元数据中对应变量的变量值,生成代码文件,其中,生成的代码文件声明生成的包名。

[0121] 在一个实施例中,计算机程序被处理器执行时还实现以下步骤:判断生成的代码文件是否已经存在,若是,则将生成的代码文件和已有代码文件转换成语法树,循环执行配置的合并策略生成合并后的语法树,根据合并后的语法树生成合并后的代码文件。

[0122] 本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程,是可以通过计算机程序来指令相关的硬件来完成,计算机程序可存储于一非易失性计算机可读取存储介质中,该计算机程序在执行时,可包括如上述各方法的实施例的流程。其中,本申请所提供的各实施例中所使用的对存储器、存储、数据库或其它介质的任何引用,均可包括非易失性和/或易失性存储器。非易失性存储器可包括只读存储器(ROM)、可编程ROM(PROM)、电可编程ROM(EPROM)、电可擦除可编程ROM(EEPROM)或闪存。易失性存储器可包括随机存取存储器(RAM)或者外部高速缓冲存储器。作为说明而非局限,RAM以多种形式可得,诸如静态RAM(SRAM)、动态RAM(DRAM)、同步DRAM(SDRAM)、双数据率SDRAM(DDRSDRAM)、增强型SDRAM(ESDRAM)、同步链路(Synchlink)DRAM(SLDRAM)、存储器总线(Rambus)直接RAM(RDRAM)、直接存储器总线动态RAM(DRDRAM)、以及存储器总线动态RAM(RDRAM)等。

[0123] 以上实施例的各技术特征可以进行任意的组合,为使描述简洁,未对上述实施例中的各个技术特征所有可能的组合都进行描述,然而,只要这些技术特征的组合不存在矛盾,都应当认为是本说明书记载的范围。

[0124] 以上实施例仅表达了本申请的几种实施方式,其描述较为具体和详细,但并不能因此而理解为对发明专利范围的限制。应当指出的是,对于本领域的普通技术人员来说,在不脱离本申请构思的前提下,还可以做出若干变形和改进,这些都属于本申请的保护范围。因此,本申请专利的保护范围应以所附权利要求为准。

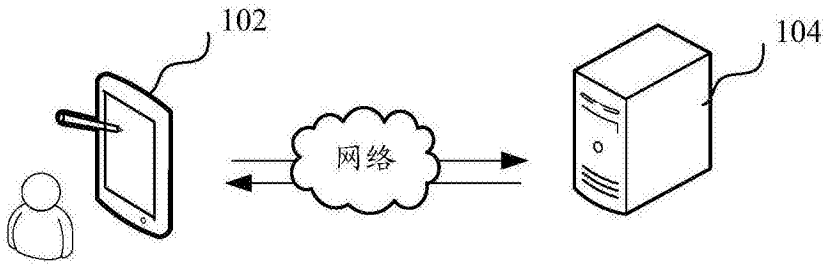

102
104
网络

图1

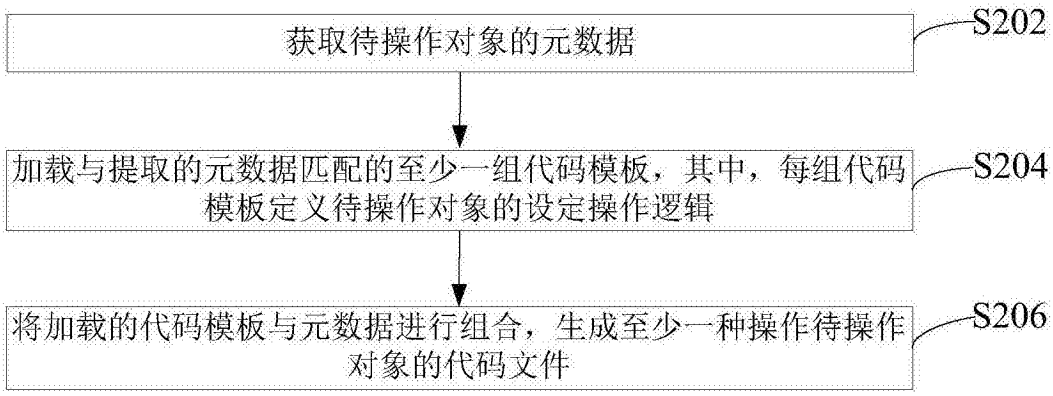

获取待操作对象的元数据
S202
加载与提取的元数据匹配的至少一组代码模板，其中，每组代码模板定义待操作对象的设定操作逻辑
S204
将加载的代码模板与元数据进行组合，生成至少一种操作待操作对象的代码文件
S206

图2

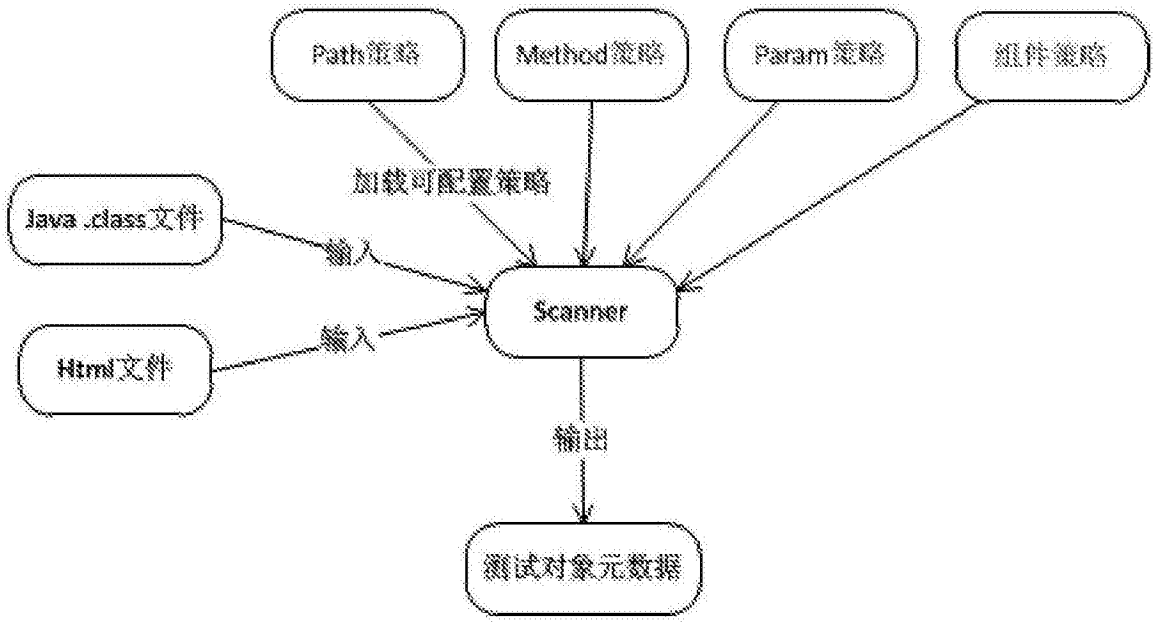

Path策略
Method策略
Param策略
组件策略
加载可配置策略
Java .class文件
输入
Scanner
Html文件
输入
输出
测试对象元数据

图3

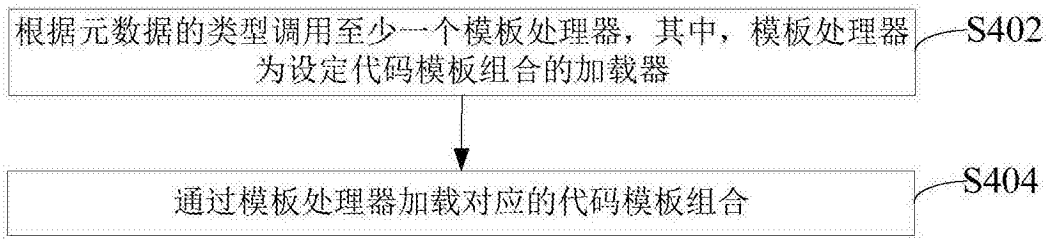

根据元数据的类型调用至少一个模板处理器，其中，模板处理器为设定代码模板组合的加载器
S402
通过模板处理器加载对应的代码模板组合
S404

图4

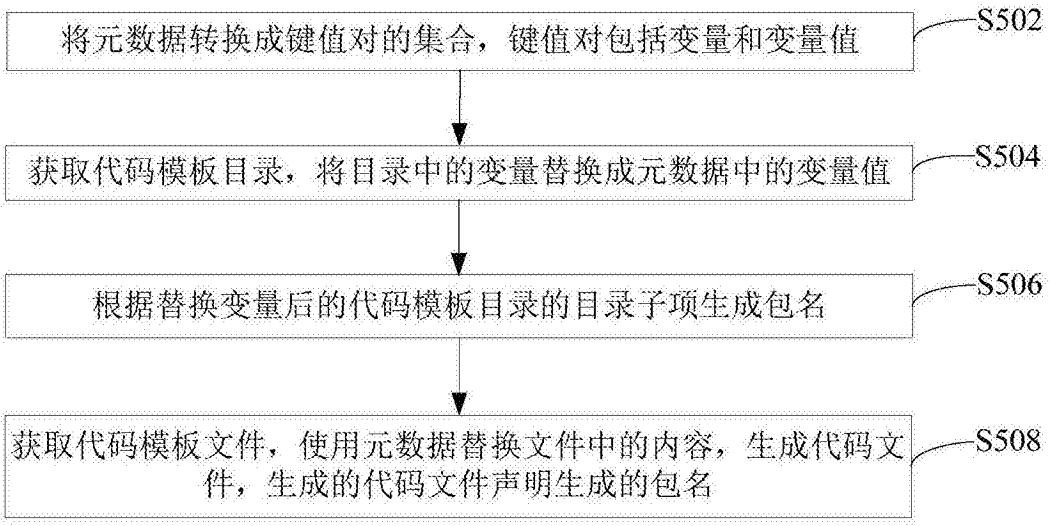

将元数据转换成键值对的集合，键值对包括变量和变量值
S502
获取代码模板目录，将目录中的变量替换成元数据中的变量值
S504
根据替换变量后的代码模板目录的目录子项生成包名
S506
获取代码模板文件，使用元数据替换文件中的内容，生成代码文件，生成的代码文件声明生成的包名
S508

图5

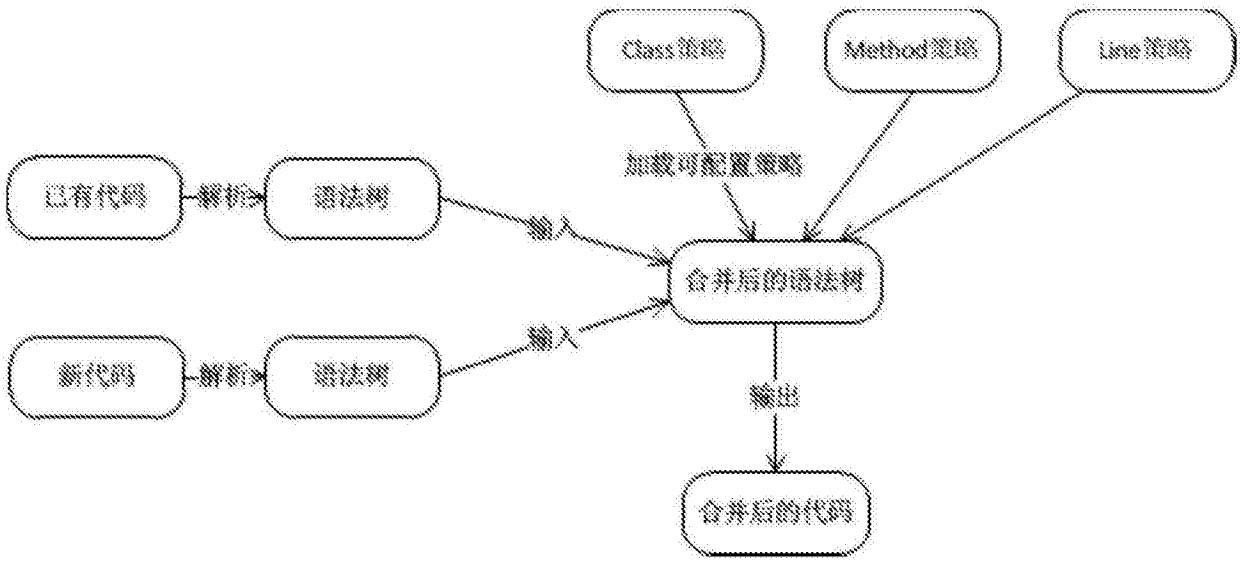


图6

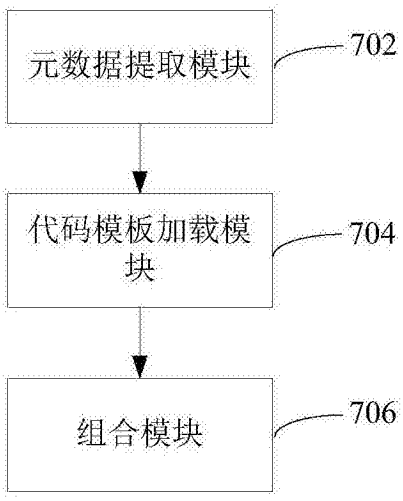

元数据提取模块
702
代码模板加载模块
704
组合模块
706

图7

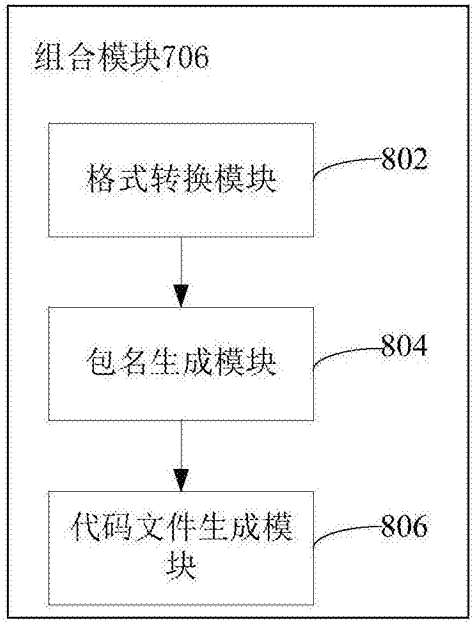

组合模块706
格式转换模块
802
包名生成模块
804
代码文件生成模块
806

图8

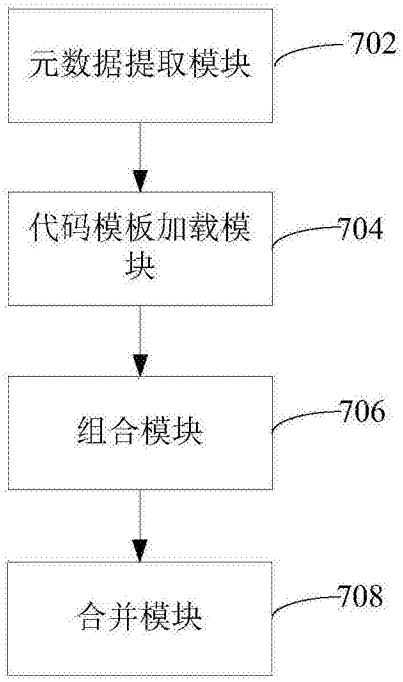

元数据提取模块
702
代码模板加载模块
704
组合模块
706
合并模块
708

图9

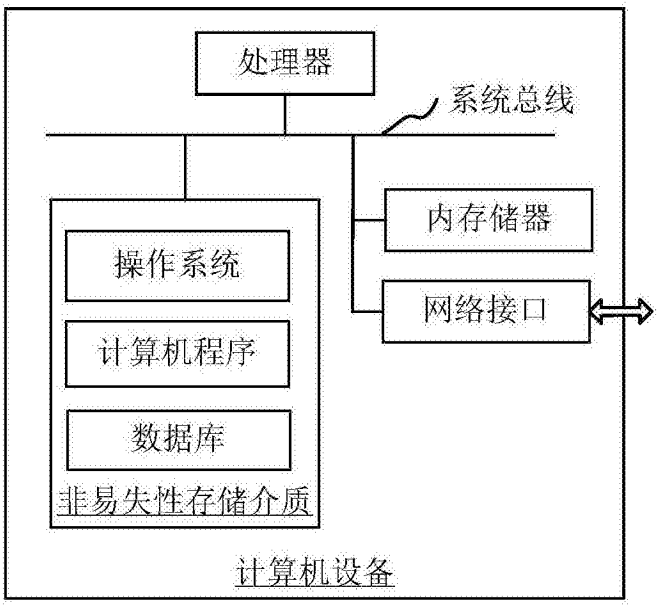

处理器
系统总线
内存储器
网络接口
操作系统
计算机程序
数据库
非易失性存储介质
计算机设备

图10