(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 102707948 A
(43)申请公布日 2012.10.03

(21)申请号 201210126647.X

(22)申请日 2012.04.26

(71)申请人 华亚微电子(上海)有限公司
地址 201203 上海市浦东新区张江高科技园区碧波路690号5号楼201-202室

(72)发明人 董琦 李知伟 王岩峰 靳文辉 付晶

(74)专利代理机构 上海智信专利代理有限公司 31002
代理人 邓琪

(51)Int.Cl.
G06F 9/44(2006.01)

权利要求书 1 页 说明书 8 页

(54)发明名称

一种源代码生成方法

(57)摘要

本发明涉及一种源代码生成方法,包括以下步骤:步骤S1,在数据库中存储字符串个例、与所述字符串个例对应的字符串通例以及与所述字符串通例对应的代码段通例,其中,所述字符串个例为字符串通例的子集;步骤S2,在所述数据库中检索与外围输入的字符串匹配的字符串个例,若检索得到该字符串个例,则执行步骤S3,否则返回执行所述步骤S1;步骤S3,在所述数据库中检索得到与所述步骤S2中获得的字符串个例对应的字符串通例,并比较该字符串个例和字符串通例,生成该字符串个例和字符串通例的第一映射关系。本发明可以使自然语言字符串自动生成代码段,从而减少软件开发工作量,满足用户的多种使用需求。

CN 102707948 A

1. 一种源代码生成方法，其特征在于，所述方法包括以下步骤：

步骤 S1，在数据库中存储字符串个例、与所述字符串个例对应的字符串通例以及与所述字符串通例对应的代码段通例，其中，所述字符串个例为字符串通例的子集；

步骤 S2，在所述数据库中检索与外围输入的字符串匹配的字符串个例，若检索得到该字符串个例，则执行步骤 S3，否则返回执行所述步骤 S1；

步骤 S3，在所述数据库中检索得到与所述步骤 S2 中获得的字符串个例对应的字符串通例，并比较该字符串个例和字符串通例，生成该字符串个例和字符串通例的第一映射关系；

步骤 S4，在所述数据库中检索得到与所述步骤 S3 中获得的字符串通例对应的代码段通例，并比较该字符串通例和代码段通例，生成该字符串通例和代码段通例的第二映射关系；

步骤 S5，根据所述第一映射关系以及所述第二映射关系，将所述步骤 S4 中获得的代码段通例替换生成源代码。

2. 根据权利要求 1 所述的源代码生成方法，其特征在于，所述步骤 S1 还包括在所述数据库中存储由所述字符串通例分解而成的多个扩展字符串通例。

3. 根据权利要求 2 所述的源代码生成方法，其特征在于，所述方法包括在所述步骤 S4 之后执行：

步骤 S6，在所述数据库中检索得到与所述步骤 S4 中获得的字符串通例对应的多个扩展字符串通例，并生成该字符串通例与多个扩展字符串通例的第三映射关系；

步骤 S7，将所述步骤 S6 中的每个扩展字符串通例作为一个字符串通例，并在所述数据库中检索与每个字符串通例对应的代码段通例，若检索得到代码段通例，则执行步骤 S8，否则，根据所述第一映射关系，将所述扩展字符串通例替换为扩展字符串个例，并将该扩展字符串个例作为字符串通例，在所述数据库中检索与该字符串通例对应的代码段通例，若检索得到代码段通例，则执行步骤 S8，否则返回执行步骤 S1；

步骤 S8，比较所述步骤 S7 中的字符串通例和获得的与其对应的代码段通例，并生成该字符串通例和代码段通例的第四映射关系；

步骤 S9，根据所述第一映射关系、所述第三映射关系以及所述第四映射关系，将所述步骤 S7 中获得的代码段通例替换生成扩展源代码；

步骤 S10，根据所述第二映射关系以及所述第三映射关系，将多个所述步骤 S9 中获得的扩展源代码嵌套进所述步骤 S4 中获得的代码段通例中，生成源代码。

4. 根据权利要求 3 所述的源代码生成方法，其特征在于，所述步骤 S6 还包括在所述数据库中存储所述第三映射关系。

5. 根据权利要求 3 或 4 所述的源代码生成方法，其特征在于，所述步骤 S8 还包括在所述数据库中存储所述第四映射关系。

6. 根据权利要求 1、2、3 或 4 所述的源代码生成方法，其特征在于，所述步骤 S3 还包括在所述数据库中存储所述第一映射关系。

7. 根据权利要求 6 所述的源代码生成方法，其特征在于，所述步骤 S4 还包括在所述数据库中存储所述第二的映射关系。

一种源代码生成方法

技术领域

[0001] 本发明涉及一种以自然语言作为输入的源代码生成方法。

背景技术

[0002] 目前，自然语言处理技术正逐步开始实用化（例如：基于自然语言的编程工具，语音控制系统等），但是仍然处于比较初级的阶段。

[0003] 现有的自然语言处理技术通常只是将一个自然语言字符串映射为一个函数。例如：将“发送短信”映射为手机内的某个函数。当然，将一个自然语言字符串映射为多个函数的序列也很容易，因为这种做法等价于将这些函数封装为一个新函数。

[0004] 上述这些实现方法的本质是：建立“字符串”和“代码段”之间的映射表，当“字符串”被输入时，执行相应的“代码段”。当然，这种映射可以是多对多的，例如：同一个“字符串”在不同应用环境下映射为不同“代码段”；多个“字符串”映射为同一个“代码段”。

[0005] 然而，这些方法存在以下的缺陷是：“代码段”是工程师预先写好的，这意味着工程师要预先判定用户会输入哪些“字符串”，由于海量的用户会有海量的差异化需求，因此，这些方法显然不符合需求的差异化趋势。

发明内容

[0006] 为了解决上述现有技术存在的问题，本发明旨在提供一种源代码生成方法，以使自然语言字符串自动生成代码段，从而减少软件开发工作量，满足用户的多种使用需求。

[0007] 本发明所述的一种源代码生成方法，包括以下步骤：

[0008] 步骤 S1，在数据库中存储字符串个例、与所述字符串个例对应的字符串通例以及与所述字符串通例对应的代码段通例，其中，所述字符串个例为字符串通例的子集；

[0009] 步骤 S2，在所述数据库中检索与外围输入的字符串匹配的字符串个例，若检索得到该字符串个例，则执行步骤 S3，否则返回执行所述步骤 S1；

[0010] 步骤 S3，在所述数据库中检索得到与所述步骤 S2 中获得的字符串个例对应的字符串通例，并比较该字符串个例和字符串通例，生成该字符串个例和字符串通例的第一映射关系；

[0011] 步骤 S4，在所述数据库中检索得到与所述步骤 S3 中获得的字符串通例对应的代码段通例，并比较该字符串通例和代码段通例，生成该字符串通例和代码段通例的第二映射关系；

[0012] 步骤 S5，根据所述第一映射关系以及所述第二映射关系，将所述步骤 S4 中获得的代码段通例替换生成源代码。

[0013] 在上述的源代码生成方法中，所述步骤 S1 还包括在所述数据库中存储由所述字符串通例分解而成的多个扩展字符串通例。

[0014] 在上述的源代码生成方法中，包括在所述步骤 S4 之后执行：

[0015] 步骤 S6，在所述数据库中检索得到与所述步骤 S4 中获得的字符串通例对应的多

个扩展字符串通例，并生成该字符串通例与多个扩展字符串通例的第三映射关系；

[0016] 步骤 S7，将所述步骤 S6 中的每个扩展字符串通例作为一个字符串通例，并在所述数据库中检索与每个字符串通例对应的代码段通例，若检索得到代码段通例，则执行步骤 S8，否则，根据所述第一映射关系，将所述扩展字符串通例替换为扩展字符串个例，并将该扩展字符串个例作为字符串通例，在所述数据库中检索与该字符串通例对应的代码段通例，若检索得到代码段通例，则执行步骤 S8，否则返回执行步骤 S1；

[0017] 步骤 S8，比较所述步骤 S7 中的字符串通例和获得的与其对应的代码段通例，并生成该字符串通例和代码段通例的第四映射关系；

[0018] 步骤 S9，根据所述第一映射关系、所述第三映射关系以及所述第四映射关系，将所述步骤 S7 中获得的代码段通例替换生成扩展源代码；

[0019] 步骤 S10，根据所述第二映射关系以及所述第三映射关系，将多个所述步骤 S9 中获得的扩展源代码嵌套进所述步骤 S4 中获得的代码段通例中，生成源代码。

[0020] 在上述的源代码生成方法中，所述步骤 S6 还包括在所述数据库中存储所述第三映射关系。

[0021] 在上述的源代码生成方法中，所述步骤 S8 还包括在所述数据库中存储所述第四映射关系。

[0022] 在上述的源代码生成方法中，所述步骤 S3 还包括在所述数据库中存储所述第一映射关系。

[0023] 在上述的源代码生成方法中，所述步骤 S4 还包括在所述数据库中存储所述第二的映射关系。

[0024] 由于采用了上述的技术解决方案，本发明具有以下优点：

[0025] 1、提高软件开发效率，降低软件开发难度

[0026] 可以将工程师设计与现有编程工具实现的协作模式下的大部分工作简化到写文档的难度，同时大部分工作能忽略细节（例如，将有向图压缩为字符串，反复进行），从而提高开发效率。

[0027] 例如，对智能手机和智能电视等产品而言，大多采用应用商店的模式。降低软件开发难度，意味着更多的人能参与应用的开发，从而获得更多的应用。

[0028] 更进一步的，可以使没有编程经验的人参与软件开发，从而获得更好的应用。例如：让医生开发和药品有关的软件。

[0029] 2、改善电子产品的用户体验

[0030] 采用本发明能根据用户的不同需求，生成不同的软件源代码，类似于为每个用户设计最适合他的软件，从而改善用户体验。

[0031] 一般的设计方法是以少数几种设计满足大多数用户的共同需求，这样做往往会忽略细节和少数用户。例如：有些人的手指比较粗。

[0032] 本发明则允许用户在购买产品后，以自然语言的方式修改一部分软件以适应自身的需求。例如：把图标放大以适应手指。

[0033] 当然，这种差异化的需求是可以通过系统设置来解决的，但是问题在于工程师并不能预先判定会有哪些差异化的需求。在本发明中，只要将数据库放在云端的服务器上，就能不断增加支持的字符串，从而支持越来越多的差异化需求。

具体实施方式

[0034] 下面结合实施例，对本发明予以详细描述。

[0035] 本发明，即一种源代码生成方法，包括以下步骤：

[0036] 步骤 S1，在数据库中存储字符串个例、与字符串个例对应的字符串通例以及与字符串通例对应的代码段通例，其中，字符串个例为字符串通例的子集；

[0037] 步骤 S2，在数据库中检索与外围输入的字符串匹配的字符串个例，若检索得到该字符串个例，则执行步骤 S3，否则返回执行步骤 S1；

[0038] 步骤 S3，在数据库中检索得到与步骤 S2 中获得的字符串个例对应的字符串通例，并比较该字符串个例和字符串通例，生成以及在数据库中存储该字符串个例和字符串通例的第一映射关系；

[0039] 步骤 S4，在数据库中检索得到与步骤 S3 中获得的字符串通例对应的代码段通例，并比较该字符串通例和代码段通例，生成以及在数据库中存储该字符串通例和代码段通例的第二映射关系；

[0040] 步骤 S5，根据步骤 S3 中生成的字符串个例和字符串通例的第一映射关系以及步骤 S4 中生成的字符串通例和代码段通例的第二映射关系，将步骤 S4 中获得的代码段通例替换生成源代码。

[0041] 在本发明的源代码生成方法中，步骤 S1 还包括：在数据库中存储由字符串通例分割而成的多个扩展字符串通例。在此情况下，本方法包括在步骤 S4 之后执行：

[0042] 步骤 S6，在数据库中检索得到与步骤 S4 中获得的字符串通例对应的多个扩展字符串通例，并生成以及在数据库中存储该字符串通例与多个扩展字符串通例的第三映射关系；

[0043] 步骤 S7，将步骤 S6 中的每个扩展字符串通例作为一个字符串通例，并在数据库中检索与每个字符串通例对应的代码段通例，若检索得到代码段通例，则执行步骤 S8，否则，根据步骤 S3 中生成的第一映射关系，将扩展字符串通例替换为扩展字符串个例，并将该扩展字符串个例作为字符串通例，在数据库中检索与该字符串通例对应的代码段通例，若检索得到代码段通例，则执行步骤 S8，否则返回执行步骤 S1；

[0044] 步骤 S8，比较步骤 S7 中的字符串通例和获得的与其对应的代码段通例，并生成以及在数据库中存储该字符串通例和代码段通例的第四映射关系；

[0045] 步骤 S9，根据步骤 S3 中生成的第一映射关系、步骤 S6 中生成的第三映射关系以及所述步骤 S8 中生成的第四映射关系，将步骤 S7 中获得的代码段通例替换生成扩展源代码；

[0046] 步骤 S10，根据步骤 S4 中生成的第二映射关系以及步骤 S6 中生成的第三映射关系，将多个步骤 S9 中获得的扩展源代码嵌套进步骤 S4 中获得的代码段通例中，生成源代码。

[0047] 基于上述步骤可知，在本发明步骤 S1 中提及的数据库中需要保存以下信息：

[0048] 1、字符串个例：例如，“100”；

[0049] 2、字符串通例：例如，“数字”；

[0050] 通例和个例的关系是一种可级联的相对关系，也可以理解为分类，个例是通例的

一个子集或特例，例如，“100”是“10 进制数字”的个例，“10 进制数字”是“数字”的个例；个例和通例是相对而言的概念，例如：“10 进制数字”相对“100”是通例，相对“数字”是个例；假如两个字符串分别包含“个例”和“通例”，且其它部分完全一致，那么这两个字符串也互为“个例”和“通例”，例如“100+100 ＝ 200”是“数字 + 数字＝数字”的个例；假如两段源代码分别包含“个例”和“通例”，且其它部分完全一致，那么这两个源代码也互为“个例”和“通例”，例如“SendMsg(“Tom”)”是“SendMsg(“人名”)”的个例；

[0051] 3、第一映射关系：例如，字符串个例“100”对应字符串通例“数字”，则第一映射关系为“数字是 100”；

[0052] 4、代码段通例：既可以是字符串形态，也可以是算法形态，而且并非一定是一行代码，也可以是多行代码，包括：预置函数、链表等算法、加法等运算以及 If 等控制语句等，例如，字符串形态的“数字＝数字”，算法形态的“输出数字”“输出＝”“输出数字”，代码段通例随源代码程序语言的变化而变化，例如：C 语言和 Java 语言对应的代码段通例是不同的；不同的软件中也有不同的代码段通例，例如：不同操作系统中获取系统时间的函数就不同；一部分“字符串通例对应的代码段通例”，其实就是某个函数的使用方法，例如：“获得系统时间字符串”对应于函数“字符串＝ GetSysTime()”；本发明说明的是一种通用的方法，而不限于具体编程语言或软件；

[0053] 5、第二映射关系：例如，字符串通例“将数字赋值给数字”对应代码段通例“数字＝数字”，则第二映射关系为“代码段通例的第 1 个数字是字符串通例的第 2 个数字，代码段通例的第 2 个数字是字符串通例的第 1 个数字”；

[0054] 6、扩展字符串通例：例如，字符串个例“如果 a 大于 0 那么 b 等于 0”对应字符串通例“如果数据大于数据那么数据等于数据”，则该字符串通例可分解为第一个扩展字符串通例“数据大于数据”和第二个扩展字符串通例“数据等于数据”；

[0055] 7、第三映射关系：例如，字符串通例“如果数据大于数据那么数据等于数据”对应第一个扩展字符串通例“数据大于数据”和第二个扩展字符串通例“数据等于数据”，则第三映射关系为“字符串通例的第 1 个数字为第一个扩展字符串通例中的第 1 个数字，字符串通例的第 2 个数字为第一个扩展字符串通例中的第 2 个数字，字符串通例的第 3 个数字为第二个扩展字符串通例中的第 1 个数字，字符串通例的第 4 个数字为第二个扩展字符串通例中的第 2 个数字”；

[0056] 8、第四映射关系：这种关系的本质与第二映射关系相同，例如，第一个扩展字符串通例“数据大于数据”作为字符串通例时对应代码段通例“数据＞数据”，则第四映射关系为“代码段通例的第 1 个数字是第一个扩展字符串通例的第 1 个数字，代码段通例的第 2 个数字是第一个扩展字符串通例的第 2 个数字”。

[0057] 本发明步骤 S2 中提及的外围输入的字符串，在本发明中默认为已经经过正确识别的自然语言字符串（自然语言的识别是指：将句子拆离为词；识别词和句子的含义；字符串的模糊匹配等分析算法），本发明仅解决如何将其替换生成源代码的问题。

[0058] 需要注意的是，数据库中的信息是可以不断添加的，使用现有编程工具的工程师在使用过程中可以不断增加信息，或者可以这样理解：向数据库添加信息＝教工具写代码；利用数据库的信息生成代码＝用工具生成代码；具体来说：

[0059] 1、当工具无法生成代码时就应该教工具写代码；由于数据库之间可以同步信息，

所以只要有一个人教会工具生成某种代码，所有的工具就都学会了。

[0060] 2、当工具生成源代码时，同时也在建立映射关系

[0061] A) 例如：完成了函数 Get_SysTime，同时向工具说明该函数的作用为“获得系统时间”；

[0062] B) 工具在“获得系统时间”和“数据＝ Get_SysTime()”之间建立对应关系，其中需要工程师输入必要的信息；

[0063] C) 换句话讲，用工具生成某个模块源代码的过程，就是为其它模块准备信息的过程。

[0064] 严格意义上来说，数据库中既可以保存“字符串个例”和“代码段个例”之间的映射关系，也可以保存“字符串通例”和“代码段通例”之间的映射关系，这并不矛盾，就像一个人既会“1+1”，也会“整数 + 整数”，还会“实数 + 实数”，而差异在于，通例化程度越高，需要保存的信息越少，因此，本发明中选择保存“字符串通例”和“代码段通例”之间的映射关系。

[0065] 下面通过举例说明上述步骤 S3- 步骤 S5。

[0066] 例如，与外围输入的字符串匹配的字符串个例为“将 data1 赋值给 data2”；

[0067] 步骤 S3，检索到与字符串个例“将 data1 赋值给 data2”对应的字符串通例“将数字赋值给数字”，同时比较两者后，生成并存储第一映射关系 M1 为：字符串通例的第 1 个数字是 data1，字符串通例的第 2 个数字是 data2（该第一映射关系 M1 来自于分析过程：data1 和 data2 被替换为数字）；

[0068] 步骤 S4，检索到与字符串通例“将数字赋值给数字”对应的代码段通例“数字＝数字”，同时比较两者后，生成并存储第二映射关系 M2 为：代码段通例的第 1 个数字是字符串通例的第 2 个数字，代码段通例的第 2 个数字是字符串通例的第 1 个数字；

[0069] 步骤 S5，根据第一映射关系 M1 和第二映射关系 M2，将代码段通例“数字＝数字”替换为源代码“data2 ＝ data1”；本实施例中，这一“替换”的过程可采用现有的编程手段实现，例如，利用华亚微电子公司设计的编程工具实现；其原理如下：

[0070] (1) 扫描代码段通例，找到其中包含的字符串通例：第 1 个“数字”和第 2 个“数字”；

[0071] (2) 由第一映射关系 M1 和第二映射关系 M2 获得：第 1 个数字是 data2，第 2 个数字是 data1；

[0072] (3) 将 data2 和 data1 代入“数字＝数字”，即，通过简单的字符串替换就可以得到源代码“data2 ＝ data1”了。

[0073] 下面通过举例说明上述步骤 S3、步骤 S4、步骤 S6- 步骤 S10。

[0074] 例如，与外围输入的字符串匹配的字符串个例为“如果 a 大于 0 那么 b 等于 0”；

[0075] 步骤 S3，检索到与字符串个例“如果 a 大于 0 那么 b 等于 0”对应的字符串通例“如果数据大于数据那么数据等于数据”，同时比较两者后，生成并存储第一映射关系 M1 为：字符串通例的第 1 个数字是 a，字符串通例的第 2 个数字是 0，字符串通例的第 3 个数字是 b，字符串通例的第 4 个数字是 0；

[0076] 步骤 S4，检索到与字符串通例“如果数据大于数据那么数据等于数据”对应的代码段通例“If(X){X}”，同时比较两者后，生成并存储第二映射关系 M2 为：代码段通例的第 1 个 X 是字符串通例中的“数据大于数据”，代码段通例的第 2 个 X 是字符串通例的“数据等

于数据”；

[0077] 步骤 S6，检索到与字符串通例“如果数据大于数据那么数据等于数据”对应的第一个扩展字符串通例“数据大于数据”和第二个扩展字符串通例“数据等于数据”，同时生成并存储第三映射关系 M3 为：字符串通例的第 1 个数字为第一个扩展字符串通例中的第 1 个数字，字符串通例的第 2 个数字为第一个扩展字符串通例中的第 2 个数字，字符串通例的第 3 个数字为第二个扩展字符串通例中的第 1 个数字，字符串通例的第 4 个数字为第二个扩展字符串通例中的第 2 个数字；

[0078] 步骤 S7，检索到第一个扩展字符串通例“数据大于数据”作为字符串通例时对应的代码段通例“数据＞数据”，检索到第二个扩展字符串通例“数据等于数据”作为字符串通例时对应的代码段通例“数据＝数据”；

[0079] 步骤 S8，比较第一个扩展字符串通例“数据大于数据”和与其对应的代码段通例“数据＞数据”后，生成并存储第四映射关系 M4 为：代码段通例的第 1 个数字是第一个扩展字符串通例的第 1 个数字，代码段通例的第 2 个数字是第一个扩展字符串通例的第 2 个数字；比较第二个扩展字符串通例“数据大于数据”和与其对应的代码段通例“数据＝数据”后，生成并存储第四映射关系 M4 为：代码段通例的第 1 个数字是第二个扩展字符串通例的第 1 个数字，代码段通例的第 2 个数字是第二个扩展字符串通例的第 2 个数字；

[0080] 步骤 S9，根据第一映射关系 M1、第三映射关系 M3 以及第四映射关系 M4，将步骤 S7 中获得的代码段通例“数据＞数据”和“数据＝数据”分别替换生成扩展源代码“a ＞ 0”和“b ＝ 0”；

[0081] 步骤 S10，根据第二映射关系 M2 以及第三映射关系 M3，将多个步骤 S8 中获得的扩展源代码嵌套进步骤 S4 中获得的代码段通例“If(X){X}”中，生成源代码“If(a ＞ 0){b ＝ 0}”。

[0082] 此处再举出下例以说明上述步骤 S7 中的另一种情况。

[0083] 例如，字符串个例为“订购 10 苹果”，检索到的字符串通例为“订购重量斤货物”，检索到由字符串通例“订购重量斤货物”分解形成的 3 个扩展字符串通例：“查询货物的单价”、“总价＝单价＊重量”和“支付总价”，在这里，“10”是“重量”的个例，“苹果”是“货物”的个例；

[0084] 在检索 3 个扩展字符串通例对应的代码段通例过程中，无法检索到与扩展字符串通例“查询货物的单价”对应的代码段通例，此时就尝试将该扩展字符串通例内部的各个“通例”反向替换为“个例”，即替换为扩展字符串个例来查找代码段通例（各种组合形式中只要有一个能找到即可），此处，将扩展字符串通例“查询货物的单价”替换为扩展字符串个例“查询苹果的单价”，并检索到对应的代码段通例“单价＝ GetApplePrice()”；其他的两个扩展字符串通例“总价＝单价＊重量”和“支付总价”则分别对应代码段通例“总价＝单价＊重量”和“Pay（总价）”；

[0085] 此处需要注意的是，在上述代码段通例中出现的两个特殊的词：“单价”和“总价”，在将代码段通例替换为源代码的过程，也需要将这两个词替换，然后在原始的字符串个例为“订购 10 苹果”中并没有出现这两个词，因此，这种情况下就需要自动生成一个不重复的临时字符串，作为临时变量，其类型可以由使用的地方确定，例如函数 GetApplePrice 的返回值类型；临时字符串指：将字符串通例分解为多个扩展字符串通例后出现的字符串，例

如此处的“单价”和“总价”，定义临时字符串的信息也保存于第三映射关系中；不重复是指源代码中的变量名不能重复，例如：前缀 + 递增的数值，ABCD_00000000001，生成源代码时，字符串“定义单价”对应于“int ABCD_00000000001”。代码段通例中所有用到“单价”的地方均替换为 ABCD_00000000001。

[0086] 又，如之前所述，本发明中的“个例”和“通例”之间是可以多层级联的，所以可以通过多层级联减少需要维护的映射关系数量。例如上例中可以认为，“重量”、“单价”、“总价”都是“数字”的个例，因此，扩展字符串通例“总价＝单价＊重量”可以作为一个字符串个例，来替换成字符串通例“数字＝数字＊数字”，也就是说，这一字符串通例“数字＝数字＊数字”能解决所有乘法的问题，因此用户并不需要维护“重量”、“单价”、“总价”等字符串之间的乘法关系。

[0087] 需要注意的是，当同一个字符串多次出现时需要做区分，例如上例中“数字”出现了 3 次，其中一种区分方法的可以是在分析过程中将字符编码长度扩展几个字节（例如从 Unicode 的 2 字节编码格式扩展为 4 字节），然后在扩展部分存储偏移量，分析结束后再转换回原有编码格式，这些都是现有技术中常规的技术手段。

[0088] 本发明中，可以将字符串通例分解为几个扩展字符串通例的链式序列，实际上也可以分解为类似于流程图的有向图序列，例如：“如果 ***，那么 ***，否则 ***”；有向图中的每个结点是扩展字符串通例，并且每个扩展字符串通例都可以分解为更加细节的有向图，或者对应着某个代码段通例。有向图的链接关系则对应着 if、for、switch 等选择性的代码段通例，例如：“if(***){***}else{***}”。当然，这种有向图序列和流程图还是有本质差异的：通过现有的编程工具可以根据这种有向图序列生成源代码，而软件中的流程图需要工程师手动编写源代码。

[0089] 由此看来，本方法能够将复杂的有向图压缩为一个字符串，其实质是信息的压缩，且这个压缩过程可以反复进行，等价于 if 等控制语句的反复嵌套，从而将信息压缩到极限，对使用“字符串通例”的人而言，完全可以忽略其中包含的细节性信息。

[0090] 另外，本发明的步骤中还提及了返回步骤 S1 的情况，这是因为一开始就在数据库中找不到字符串个例，即，说明数据库缺少信息，报告用户要求添加信息即可；或者在字符串通例分解的过程中，找不到对应的代码段通例，从而导致生成源代码失败；后者是个可回退的分析过程，即，字符串引入通例和个例的映射关系后存在很多不同的解释，分析过程是逐渐尝试各种可能，直至有一种成功或全部失败，假如失败，则也说明数据库缺少信息，报告用户要求添加信息即可。

[0091] 基于上述说明可知，本方法应用的前提是建立数据库，这个数据库可以保存在云端的服务器上，保存字符串间的个例和通例关系，字符串通例对应的代码段通例，字符串通例间的扩展型映射关系。而并不需要预先判断有哪些字符串，相关的信息可以在运行过程中不断添加，换句话说：使用者并不需要猜测使用者会输入哪些字符串，只是在出现工具无法处理的字符串时添加这个字符串，并且任何一个使用者都可以添加他需要的字符串。并且由于信息库之间可以同步信息，所以只要有一个使用者输入了某个字符串所有人都可以用。这看似巨大的工作量其实是由所有的使用者分担的，使用者越多，每个人分到的工作量越小。

[0092] 本发明可以有以下应用模式：

[0093] 模式 1、用户是工程师，例如，用于开发自动化编程工具。

[0094] 在这种模式下，可利用现有编程工具根据工程师输入的字符串生成源代码。而软件开发中有一部分工作是现有编程工具无法自动完成的（例如视频编解码算法等），则这个部分的工作仍然由工程师手动完成，因此，这种模式其实是工程师和现有编程工具协作完成软件的过程，即，工程师设计，工具实现。

[0095] 例如：工程师指定某个函数的作用为“向链表 p_chain 插入新结点 p_node”（这是字符串个例，所以含有 p_chain 等），检索到字符串通例“向链表字符串插入新结点字符串”，就根据通例对应的代码段通例生成源代码，再将代码段通例中的字符串替换为 p_chain 等。

[0096] 这种模式下，工具的使用者可以分类两种，资深工程师负责输入字符串通例、代码段通例以及各种它们之间的对应关系，通过信息库之间同步信息，所有工程师都可以使用这些信息；普通工程师负责输入字符串个例。对普通工程师而言，他完全可以在不了解技术细节的情况下完成软件开发，从而降低工作难度。

[0097] 模式 2、用户是普通消费者，例如，用于实现增强语音控制的分析算法。

[0098] 在这种模式下，可利用算法根据用户输入的字符串生成源代码，用户输入字符串的途径可以是语音识别。当然，由于没有工程师协助，这种模式并不适用于复杂软件，只能基于“预置函数”拼接出适合用户需求的功能，通常体现为应用软件，此处的“预置函数”是指工程师预先编写的软件模块中所包含的函数，这些模块被预置在产品中，例如：数据库、文件系统等等。

[0099] 例如：用户要求“播放第 10 个电视频道”（这是字符串个例），根据算法检索到字符串通例“播放第数字个电视频道”，然后扩展为“从数据库获得第数字个电视频道的编号”和“根据编号播放电视频道”，然后替换为源代码。

[0100] 这种模式下生成的代码的编程语言应该选用解释型语言，例如Action Script等。同时数据库要保存在云端的服务器中，以便不断增加所能支持的字符串。

[0101] 例如：用户向手机说“屏蔽所有来自上海的电话”，这个字符串被发送回手机公司的云端服务器，服务器尝试根据字符串修改接电话模块。假如目前还不支持，则返回失败并向维护工程师报告。维护工程师向数据库添加相关信息后，再有这一类指令被发送到服务器，就能自动生成源代码并发送回手机了。

[0102] 本发明和下载应用程序是有差别的：本发明中用户仅仅以自然语言陈述自己的需求即可，既不需要查找应用程序，也不需要下载和安装。本发明更适用于微小或细节的修改。

[0103] 模式 2 可以和模式 1 协作，即工程师在模式 1 下完成“预置函数”，普通消费者在模式 2 下基于“预置函数”拼接出自己喜欢的功能。

[0104] 以上所述的，仅为本发明的较佳实施例，并非用以限定本发明的范围，本发明的上述实施例还可以做出各种变化。即凡是依据本发明申请的权利要求书及说明书内容所作的简单、等效变化与修饰，皆落入本发明专利的权利要求保护范围。本发明未详尽描述的均为常规技术内容。