(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 110554860 B
(45)授权公告日 2021.03.12

(21)申请号 201910804907.6

(22)申请日 2019.08.29

(65)同一申请的已公布的文献号
申请公布号 CN 110554860 A

(43)申请公布日 2019.12.10

(66)本国优先权数据
201910568216.0 2019.06.27 CN

(73)专利权人 北京大学
地址 100871 北京市海淀区颐和园路5号北京大学

(72)发明人 邹艳珍 伍仕骏 沈琦 谢冰

(74)专利代理机构 北京君尚知识产权代理有限公司 11200
代理人 司立彬

(51)Int.Cl.
G06F 8/30(2018.01)
G06F 8/41(2018.01)

(56)对比文件
CN 108388425 A,2018.08.10
CN 105739981 A,2016.07.06
US 2008134142 A1,2008.06.05
CN 107506414 A,2017.12.22
养有道.eclipse智能提示及快捷键.《CSDN》.2014,第1页.
宫爱爱.基于Eclipse智能代码生成框架的研究.《现代电子技术》.2013,

审查员 刘艳萍

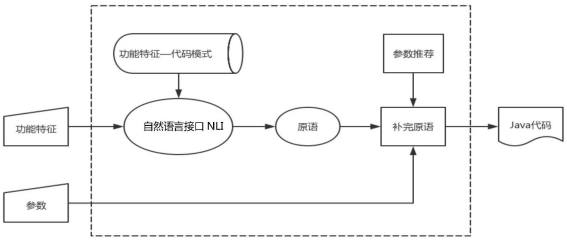
权利要求书1页 说明书6页 附图1页

(54)发明名称

一种软件项目自然语言编程接口NLI的构造方法及代码生成方法

(57)摘要

本发明公开了一种软件项目自然语言编程接口NLI的构造方法及代码生成方法。其中代码生成方法为:1)将软件项目的每一<功能特征—API调用模式>封装为NLI中的一条原语,包括:原语的功能特征描述、API调用模式、宾语参数和其他参数;2)确定原语对应的抽象语法树节点类型;每个节点类型中规定了该类型节点拥有的子节点及属性;将节点的各子节点、属性与对应的Java代码文本区域绑定后,进行NLI原语参数补全过程中,投影编辑器生成代码的抽象语法树;3)从抽象语法树的根节点开始,递归式地对各节点进行转换,将抽象语法树节点中的属性与子节点安插至其API调用模式中空缺的部分,最终,生成原语对应的Java代码。



CN 110554860 B

1.一种软件项目自然语言编程接口NLI构造方法,其步骤包括:

将选取的每一二元组封装为面向复用的软件项目自然语言编程接口中的一条原语;其中每条原语包括:原语的功能特征描述、原语的API调用模式、原语的宾语参数和原语的非宾语参数;所述二元组包括功能特征及对应的API调用模式;

原语的功能特征描述是指以自然语言动宾短语形式描述软件项目能够支持实现的一项软件功能,用于当开发者输入当前开发任务的自然语言描述后根据输入字符串推荐匹配的原语;

原语的API调用模式为所述二元组中的API调用模式,以文本形式存储;

原语的宾语参数是指从原语的功能特征描述中的动宾短语中提及的参数,

原语的非宾语参数是指宾语参数外需要填写的参数,每一参数占用一行,包含需要填写的参数及参数的用途说明;其中基于自然语言编程接口NLI对所述原语中需要填写的参数进行补全,其方法为:首先获取原语所在的二元组对应的API使用示例代码,将API使用示例代码解析成抽象语法树,并对其进行符号解析;然后提取API使用示例代码中的参数组合,并追溯每一个参数的创建路径;然后统计参数组合在API使用示例代码中的出现频数,推荐出现最为频繁的参数组合给开发者。

2.一种基于软件项目自然语言编程接口NLI的代码生成方法,其步骤包括:

1)将软件项目的每一二元组封装为自然语言编程接口NLI中的一条原语;其中所述原语包括:原语的功能特征描述、原语的API调用模式、原语的宾语参数和原语的非宾语参数;所述二元组包括功能特征及对应的API调用模式;原语的非宾语参数是指宾语参数外需要填写的参数;原语的功能特征描述是指以自然语言动宾短语形式描述软件项目能够支持实现的一项软件功能,用于当开发者输入当前开发任务的自然语言描述后根据输入字符串推荐匹配的原语;原语的宾语参数是指从原语的功能特征描述中的动宾短语中获取的宾语参数;

2)确定原语对应的抽象语法树节点类型;其中每个节点类型中都规定了该类型节点拥有的子节点以及属性;将节点的各子节点、属性与该节点对应的Java代码文本区域绑定后,在开发者进行NLI原语参数补全过程中,投影编辑器可自动调整生成代码的抽象语法树;

3)从抽象语法树的根节点开始,递归式地对抽象语法树各节点进行转换,将抽象语法树节点中的属性与子节点安插至对应节点的API调用模式中空缺的部分。

3.如权利要求2所述的方法,其特征在于,利用MPS的投影编辑器作为编辑环境,投影编辑器内部维护了程序代码的抽象语法树。

4.如权利要求2所述的方法,其特征在于,对所述原语中需要填写的参数进行补全,然后进行步骤2);其中补全方法为:首先搜集原语所在的二元组对应的使用示例代码,将API使用示例代码解析成抽象语法树,并对其进行符号解析;然后提取API使用示例代码中的参数组合,并追溯每一个参数的创建路径;然后统计参数组合在API使用示例代码中的出现频数,推荐出现最为频繁的参数组合给开发者。

5.如权利要求2所述的方法,其特征在于,维护一个不可用名字表,用于记录已用的中间变量;步骤3)的转换过程中在每次需要生成一个中间变量时通过查询该不可用名字表确保待生成中间变量未被使用过。

一种软件项目自然语言编程接口NLI的构造方法及代码生成方法

技术领域

[0001] 本发明涉及一种自动构造软件项目自然语言编程接口(Natural Language Interface,NLI)的方法和基于NLI的代码生成方法,属于计算机软件技术领域。

背景技术

[0002] 开源软件的快速发展为软件复用积累了大量的可复用资源,使得开发过程中复用的比例大幅提高,支持和推动了软件复用的发展。软件开发者在开发过程中经常需要复用已有的代码库/软件项目,利用其提供的编程接口(Application Programming Interface,API)来完成开发任务编码。

[0003] 在基于API进行软件复用的过程中,复用的对象虽然是软件API,但实际的开发任务则是围绕着需要实现的软件功能展开的。开发者经常遇到两类问题:(1)开发任务和API功能之间存在抽象层次的鸿沟。通常情况下,一个开发任务需要组合调用多个APIs来完成。因此开发者首先要检索定位到可以完成开发任务功能需求的一组APIs及其使用模式。(2)开发者需要学习并了解API的调用规约。在API调用过程中,开发者受制于严格的API调用规约,任何一个参数、符号或者调用顺序的错误都将导致软件的非正常运行甚至出错。大量统计结果表明,开发者在复用过程中往往要耗费大量的时间来查找API、学习如何调用API并进行相应的程序编码!而随着软件规模的扩大,软件API的数量不断增加,用户查找、调用API进行特定任务编码的难度也在不断增加。譬如,开源项目Lucene包含了超过5,377个类,以及34,042个方法,开发者要进行快速有效的API检索与调用无疑是非常困难的。

[0004] 为了解决上述问题,一些研究工作提出通过挖掘代码库的<功能特征-API调用模式>对来组织代码库,其中功能特征是对一项软件功能的动宾短语描述,API调用模式是调用相关API实现此项功能特征的代码片段的模式。由于挖掘得到的软件项目/代码库<功能特征-API调用模式>数量很大,开发者首先通过检索功能特征的方式找到开发者需要的API调用模式,然后填写API调用模式中的空缺部分以取得能够完成开发任务的代码片段。在此过程中,要让软件开发者能够正确补全API调用模式进行编码,开发者需要对该API调用模式有深入的认识:为了将API调用模式作用在新的上下文中,开发者需要理解该API调用模式的运行逻辑,用上下文中的变量补充到API调用模式中的空缺当中;为了将多个API调用模式组合起来,开发者需要知道API调用模式的正确组合方式,需要知道位于不同API调用模式的空缺之间的关系;如果在改编过程中遇到了不熟悉的API,还需要进一步查阅相关文档,理解该API的功能以及所需要的参数,正确地创建并填入参数。这一过程需要开发者花费大量的时间和精力,尤其对于是编程新手以及不熟悉该代码库的开发人员而言。

发明内容

[0005] 针对直接补全API调用模式的方式进行复用仍需花费开发者大量时间的问题,本发明的目的在于提出软件项目自然语言编程接口(Natural Language Interface,NLI)的

概念,并提供一种基于软件项目自然语言编程接口(NLI)的代码生成方法。通过本发明提供的方法和系统,可以将软件项目的API以及API调用模式以一种更为简洁直观的自然语言编程接口的形式表现出来,隐藏不需要开发者关心的内容(不需要开发者了解的参数、调用语法等),从而减轻开发者的负担,提高软件复用的效率。

[0006] 本发明的技术方案为:

[0007] 一种软件项目自然语言编程接口NLI构造方法,其步骤包括:

[0008] 将选取的每一对<功能特征—API调用模式>封装为面向复用的软件项目自然语言编程接口中的一条原语;其中每条原语包括:原语的功能特征描述、原语的API调用模式、原语的宾语参数和原语的其他参数;

[0009] 原语的功能特征描述是指以自然语言动宾短语形式描述软件项目能够支持实现的一项软件功能,用于当开发者输入当前开发任务的自然语言描述后根据输入字符串推荐匹配的原语;

[0010] 原语的API调用模式为<功能特征—API调用模式>中的API调用模式,以文本形式存储;

[0011] 原语的宾语参数是指从原语的功能特征描述中的动宾短语中获取的宾语参数,

[0012] 原语的其他参数是指宾语参数外需要填写的若干参数,每一参数占用一行,包含需要填写的参数及参数的用途说明。

[0013] 进一步的,基于自然语言编程接口NLI对所述原语中需要填写的若干参数进行补全,其方法为:首先获取原语所在的<功能特征—API调用模式>对应的API使用示例代码,将API使用示例代码解析成抽象语法树,并对其进行符号解析;然后提取API使用出示例代码中的参数组合,并追溯每一个参数的创建路径;然后统计参数组合在API使用示例代码中的出现频数,推荐出现最为频繁的参数组合给开发者。

[0014] 一种基于软件项目自然语言编程接口NLI的代码生成方法,其步骤包括:

[0015] 1)将软件项目的每一<功能特征—API调用模式>封装为自然语言编程接口NLI中的一条原语;其中所述原语包括:原语的功能特征描述、原语的API调用模式、原语的宾语参数和原语的其他参数;

[0016] 2)确定原语对应的抽象语法树节点类型;其中每个节点类型中都规定了该类型节点拥有的子节点以及属性;将节点的各子节点、属性与该节点对应的Java代码文本区域绑定后,在开发者进行NLI原语参数补全过程中,投影编辑器调整生成代码的抽象语法树;

[0017] 3)从抽象语法树的根节点开始,递归式地对抽象语法树各节点进行转换,将抽象语法树节点中的属性与子节点安插至其API调用模式中空缺的部分,生成原语对应的Java代码。

[0018] 进一步的,利用MPS的投影编辑器作为编辑环境,投影编辑器内部维护了程序代码的抽象语法树。

[0019] 进一步的,对所述原语中需要填写的若干参数进行补全,然后进行步骤2);其中补全方法为:首先搜集原语所在的<功能特征—API调用模式>对应的使用示例代码,将API使用示例代码解析成抽象语法树,并对其进行符号解析;然后提取API使用出示例代码中的参数组合,并追溯每一个参数的创建路径;然后统计参数组合在API使用示例代码中的出现频数,推荐出现最为频繁的参数组合给开发者。

[0020] 进一步的，维护一个不可用名字表，用于记录已用的中间变量；步骤3)的转换过程中在每次需要生成一个中间变量时通过查询该不可用名字表确保待生成中间变量未被使用过。

[0021] 本发明的方法流程如图1所示，主要包括以下三部分：

[0022] (一)自然语言编程接口NLI的构造

[0023] 基于挖掘得到的软件项目/代码库的<功能特征—API调用模式>，将其封装形成面向复用的软件项目自然语言编程接口NLI。具体的，每一对<功能特征—API调用模式>构造为NLI的一条原语。每条原语由以下几部分组成：

[0024] 1)原语的功能特征描述。每一对<功能特征—API调用模式>中的功能特征部分，以自然语言动宾短语形式描述了软件项目能够支持实现的一项软件功能。在NLI中，这个动宾短语被以字符串的形式保存下来。当开发者输入当前开发任务的自然语言描述后，系统可根据字符串匹配推荐适当的NLI原语。

[0025] 2)原语的API调用模式。对应每一对<功能特征—API调用模式>中的API调用模式部分。在NLI中，每个API调用模式也是以文本形式存储的。

[0026] 3)原语的宾语参数。原语的功能特征描述(动宾短语)中的宾语部分通常是需要开发者填入的重要参数。本发明通过自然语言词法解析获得这一参数，并进行独立描述。此外，宾语参数也可能是一个需要开发者进行声明的变量。声明之后，开发者能够在下文中继续使用这个变量。

[0027] 4)原语的其他参数。除了宾语参数外，原语中还需要填写的其他参数被一个大括号包裹起来，其中每一行对应于一个参数。在每一行中，左边给出了该参数的用途说明(可以从API调用模式中对应参数的名字和类型信息、注释信息中抽取而来)，右侧给出了开发者需要填入的具体参数。开发者需要填入的具体参数可能是一个表达式或者是一个语句，表达式参数只占一行，语句参数可占多行。

[0028] 表1一个实例对应的Java代码、API调用模式以及NLI原语

[0029]

CellStyle style = workbook.createCellSytle();
style.setFillForegroundColor(IndexedColors.RED.getIndex());
style.setFillBackgroundColor(IndexedColors.BLUE.getIndex());
style.setFillPatterhn(FillPatternType.SOLID_FOREGROUND);
cell.setCellStyle(style);
	开源软件项目POI的功能特征“set cell color”对应的Java代码

CellStyle ___ = ___.createCellSytle();
___.setFillForegroundColor(___);
___.setFillBackgroundColor(___);
___.setFillPatterhn(___);
___.setCellStyle(___);
	开源软件项目POI的功能特征“set cell color”对应的API调用模式

set color for cell <cell> {
background color : <IndexedColors.RED.getIndex()>
foreground color : <IndexedColors.BLUE.getIndex()>
fill pattern : <FillPatternType.SOLID_FOREGROUND>
}
	开源软件项目POI的功能特征“set cell color”对应的NLI原语

[0030] (二)NLI原语参数的补全

[0031] 如上文所述，NLI原语中存在许多需要开发者补充完整的参数。在本发明中，原语

的参数有必填和选填之分，每个必填部分由红色标记，选填部分则由灰色标记。

[0032] 为了帮助开发者更为方便快捷地补全这些参数，本发明对这些空缺的参数进行了关联关系分析，给出了一种参数自动补全和推荐策略。其基本思想是通过从GitHub上搜集<功能特征—API调用模式>中出现的API的使用示例代码，找到经常被使用到的参数组合推荐给开发者。除此之外，推荐机制也会考虑到参数之间存在的固定关系，从而引导开发者正确地创建并填入参数。具体过程是：

[0033] 首先，解析<功能特征—API调用模式>中API的使用示例代码。使用开源工具JavaParser将API使用示例代码解析成抽象语法树，并对其进行符号解析。

[0034] 其次，提取出示例代码中API调用模式的参数组合，并为参数组合中每一个参数追溯其创建路径。在追溯过程中，找到每个参数的定义位置，用本地定义的初始化表达式代替所有该参数出现的位置。该追溯过程递归进行，直到不存在未定义的符号。

[0035] 最后，统计参数组合在API使用示例代码中的出现频数，推荐出现最为频繁的参数组合给开发者。

[0036] (三)Java代码的生成

[0037] 在具体应用时，开发者基于任务需求定位到一个自然语言变成接口NLI后，本发明可以辅助开发者进行NLI原语参数的补全，并将补全后的NLI原语自动转换为Java代码。为此，需要实现一个代码转换器。编写代码转换器是一件过程繁琐的工作，需要编写相应的词法分析、语法分析、语义分析、代码生成等复杂模块。JetBrains公司的元编程系统MPS(Meta Programming System)对这些模块做了统一集成，并提供了用户友好的开发界面，因此本发明使用MPS开发了NLI原语到Java代码的转换器。转换器使用MPS的投影编辑器(Projectional Editor)作为编辑环境，投影编辑器内部维护了程序代码的抽象语法树。具体方法是：

[0038] 1)确定原语对应的抽象语法树(Abstract Syntax Tree,AST)节点类型。本发明将每一条原语都定义为一种新的AST节点类型。每个节点类型中都规定了该类型节点可以拥有的子节点以及属性。在将节点类型的各子节点、属性与该节点对应的Java代码文本区域绑定后，在开发者进行NLI原语参数补全过程中，投影编辑器将会自动调整生成代码的抽象语法树。AST节点在抽象语法树中的位置根据程序的结构确定。

[0039] 2)将NLI原语转换为Java代码。该转换过程本质是将原语对应的抽象语法树节点中的属性与子节点安插至其API调用模式中空缺的部分。转换过程从抽象语法树的根节点开始，递归式地对抽象语法树各节点进行转换。转换过程中必须确保名字的唯一性，因此本发明会维护一个不可用名字表，在每一次需要生成一个中间变量时确保它的名字没有被使用过。

[0040] 与现有技术相比，本发明的积极效果为：

[0041] 本发明通过构造软件项目自然语言接口(NLI)，为开发者复用软件项目提供了简单、便捷且易于理解的编程方式。NLI屏蔽了开发者不需要关心的细节，并推荐其中每个空缺参数的填充方式，由此辅助开发者快速生成开发任务对应的Java代码。

附图说明

[0042] 图1是本发明的总体框架图。

具体实施方式

[0043] 为使本发明的上述特征和优点能更明显易懂,下文特举实施例,详细说明如下:

[0044] POI是Apache社区的一个著名开源软件项目。表2展示了这个软件项目的30项功能特征列表。这些动宾短语形式的功能特征描述来自POI的官方文档,且每条功能特征都有对应的API调用模式。

[0045] 表2软件项目Apache POI中的30项功能特征

[0046]

功能特征	说明
create a new HSSF workbook	创建一个HSSF格式的工作簿
create a cell block for a sheet	在表格中创建一个单元格的块
create an image in a sheet	在表格中创建一个图片
create a shape in a sheet	在表格中创建一个图形
create a textbox in sheet	在表格中创建一个文本框
fit sheet to one page	将表格调整为一页大小
set all columns as autoSize	将所有列设置为宽度自动增长
add hyperlink in a cell	在单元格内添加超链接
get cell content(plain text)	以文本形式获得单元格内容
get workbook content(plain text)	以文本形式获得工作簿内容
hide a row	隐藏一行
iterate a column	遍历一列
iterate a row	遍历一行
iterate a sheet	遍历一个表格
iterate a workbook	遍历一个工作簿
merge cells	将多个单元格合并
read a workbook from file	在文件中读取一个工作簿
save workbook in a file	保存工作簿至文件
set cell alignment	对齐单元格
set cell borders	设置单元格边框
set cell color	设置单元格颜色
set cell font	设置单元格字体
set sheet footer	设置表格页脚
set sheet header	设置表格页眉
set page number on footer	在表格页脚设置页号
set sheet print area	设置表格打印范围
set zoom magnification of sheet	设置表格缩放倍率
shift some rows in a sheet	移动表格中的行
get cell content	获得单元格内容
unhide a row	隐藏一行

[0047] 本发明将上述POI项目的30项功能特征全部封装为NLI原语。在此基础上,假设开发者需要复用该软件项目,实现一项功能为“为表格设置单元格颜色(set cell color)”的

开发任务。开发者在编辑器中敲下单词“set”并按下补全快捷键之后，编辑器中将出现一个候选列表，列表中每一项都是表2中包含“set”的一项功能特征。

[0048] 针对开发任务，开发者在列表中选定了“set cell color”功能后，编辑器中出现了对应的自然语言编程接口NLI。用户可以清楚的了解该原语的功能特征、必须要填写的参数和可选参数。

[0049] 在基础上，开发者可以首先将宾语参数中填入目标单元格，之后再根据参数推荐正确填充了背景与前景颜色、填充方式，补全了该原语。

[0050] 最后，开发者在编辑器中按下鼠标右键，在列表中选择“Preview Generated Text”之后，工具将这段程序自动转化为Java代码。

[0051] 以上实施仅用以说明本发明的技术方案而非对其进行限制，本领域的普通技术人员可以对本发明的技术方案进行修改或者等同替换，而不脱离本发明的精神和范围，本发明的保护范围应以权利要求书所述为准。

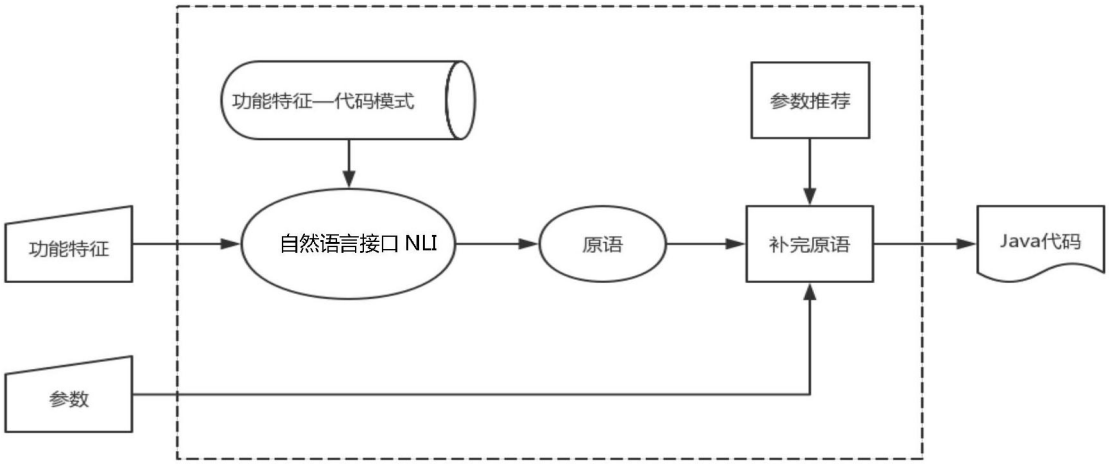


图1