肖英剑,揭安全,李宏伟,等. 一种增强代码理解的代码可视化工具 JavaCity [J].江西师范大学学报(自然科学版),2023,47(6):652-660.

XIAO Yingjian,JIE Anquan,LI Hongwei,et al. JavaCity:a code visualization tool to enhance code understanding [J].Journal of Jiangxi Normal University (Nature Science),2023,47(6):652-660.

文章编号:1000-5862(2023)-06-0652-09

一种增强代码理解的代码可视化工具 JavaCity

肖英剑[1,2],揭安全[1*],李宏伟[1*],钟崇文[1],罗 荣[3]

(1.江西师范大学计算机信息工程学院,江西 南昌 330022;2.南昌理工学院计算机信息工程学院,江西 南昌 330044;
3.江西师范大学数字产业学院,江西 上饶 334000)

摘要:该文提出了一种基于城市隐喻的沉浸式代码可视化工具 JavaCity,实现了方法级别的细粒度可视化;通过抽取软件项目各项度量值构建可视化可交互的虚拟城市,进一步增强代码可视化和代码理解的能力.在 JavaCity 和 IDEA 上进行的对比实验研究表明:JavaCity 在理解软件项目相关的任务上时间效率节省 200 s 以上,在有用性评价上接近 100%正确,近 90%的参与者在脑力劳动、努力程度和挫折程度方面认为认知负荷更低.

关键词:代码可视化;城市隐喻;代码理解;虚拟现实

中图分类号:TP 391 **文献标志码**:A **DOI**:10.16357/j.cnki.issn1000-5862.2023.06.13

0 引言

在软件生命周期中,诸如修复代码缺陷、代码重构、代码审查等软件开发和维护的活动均与代码理解息息相关.随着信息技术的发展,软件系统的规模和复杂性也在逐渐增加,开发人员需花费大量的时间来理解项目代码[1].对于新入职的开发人员而言,阅读他人编写的代码、厘清软件项目的复杂结构、理解在项目中代码的功能和作用都需要很长时间才能完成.代码理解是一个困难的过程,常常让人感到疲劳和乏味.

软件工程和可视化领域的经验证实,软件系统的可视化表示可以增强其可理解性并降低其开发成本.使用文本编辑器阅读代码难以充分理解代码,并获取源代码的一些信息(比如代码结构、复杂性等)[2].研究者们试图通过代码可视化的方式在视觉上帮助开发人员理解源代码.目前的研究工作经常运用太阳系隐喻、城市隐喻或森林隐喻的可视化技术,更加生动形象地描述软件项目,刻画代码结构和复杂性,从而为代码理解的相关任务提供支持.最近,研究人员正在探索将虚拟现实作为程序员的工具[3],由于虚拟现实(virtual reality, VR)具有更强的氛围感、空间感、距离感,当用户进入虚拟空间时,有身临其境的效果,可以更自然的方式与虚拟对象实时交互,有强烈的吸引力.但目前现有的代码可视化工具[4-5]粒度较粗(类或模块),开发人员仍需花费时间进一步理解代码,同时也缺乏可视化交互能力.

针对 Java 语言开发的软件项目,提出了一种新的代码可视化工具 JavaCity,运用虚拟现实技术和城市隐喻技术,将软件项目可视化为一座虚拟交互城市来支持代码理解活动.JavaCity 将在 Java 软件项目中的包映射为在城市中的社区,类映射成小区,方

收稿日期:2023-05-12

基金项目:国家自然科学基金(62266021),江西省自然科学基金(20224BAB202018),江西省 03 专项课题(20212ABC03A26)和江西省科技厅重大科技专项“揭榜挂帅”(20213AAE02001)资助项目.

通信作者:揭安全(1975—),男,江西广昌人,教授,主要从事智能信息处理与信息系统、教育信息化研究.E-mail:janquan@163.com

李宏伟(1975—),男,江西南康人,副教授,博士,主要从事程序分析与实证软件工程、软件知识图谱、自然语言处理研究.E-mail:lihongwei@ jxnu.edu.cn

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

法映射成建筑物.其中,建筑物的高度由代码行数决定,方法形参的数量反映在建筑物的底面积上.开发人员可以进入建筑物内,在 VR 代码墙上阅读和理解项目的源代码,开发人员使用 JavaCity 可以在代码虚拟城市中自由移动,沉浸式地探索软件项目,快速理解代码结构,从可视化软件复杂信息中获取代码的更多信息.

1　相关工作

1.1　代码理解

研究人员通过不同的方式来促进开发人员理解源代码,例如采用调用图、UML、代码摘要自动生成、软件文档等方式.调用图描述了系统及其在功能调用或操作调用方面的结构.R. Alanazi 等[6]提出了执行路径的分层聚类技术,以在不同粒度级别和不同软件单元(包括包、类和函数)下生成可视化调用图,允许开发人员通过调用图的抽象来理解大型软件系统的功能和结构.UML 图也是一种常见的理解项目的方案.UML 图对软件项目的设计和实现进行建模,描述项目的结构和行为.H.Cheers 等[7]提出一个程序分析框架,用于从应用程序源代码对序列图进行逆向工程;该框架能够从序列图中过滤出不相关的操作,使开发人员能通过 UML 序列图更容易地关注软件项目的控制流和数据流.随着深度学习技术的发展,代码摘要自动生成可以自动理解源代码的语义,许多研究者[8-10]在不断改进代码摘要生成的算法和模型,提高代码和注释的一致性.软件文档也是程序员最常见的代码理解方式,若一个软件随着维护次数的增加,变更文档可能不完整,则这将给程序理解带来挑战.S.Jiang[11]提出了使用神经机器翻译自动生成软件更改描述和软件功能描述的文档,从而增强了程序理解的能力.

1.2　代码可视化

代码可视化可以有效地增强软件系统的可理解性.代码可视化是通过将源代码转换为不同的视图来增强源代码呈现[12].早期的代码可视化研究主要是 2D 图形的形式. M. Alnabhan 等[13]提出的 2 维软件可视化技术,每个类由矩形表示,类的大小(代码行)由矩形的高度表示,方法和属性分别用圆和三角形表示,类之间的关系用箭头表示.近年来,代码可视化逐渐发展成 3D 表征形式,更具有立体感,可以给开发人员一种在系统中如同“在家”的感觉[14].U. Erra 等[15]提出了一种森林隐喻的技术,利用树干、树枝、叶子和颜色等元素映射到源代码特征上.R. Wettel 等[4]提出了一种独立于语言的交互式 3 维可视化工具 CodeCity,用于分析大型软件系统.这是最早基于城市隐喻的代码可视化,它将类描述为建筑物,将包描述为“软件城市”的区域,但仅仅是对代码结构的可视化,并且可视化粒度较粗.

1.3　VR 技术用于代码可视化

虚拟现实技术通过计算机模拟的虚拟环境及各类传感设备给使用者以身临其境的环境浸入感,并赋予使用者与虚拟物件实现信息交互的能力,具有沉浸性、交互性、构想性和感知性的特点.目前,虚拟现实技术被广泛应用于可视化中,例如 A.Hori 等[16]提出了一个名为 CodeHouse 的工具,它将源代码的结构以在 VR 空间中的房子的形式可视化.通过将代表模块的房间放置在虚拟圆柱体的内部,CodeHouse 使开发人员有可能同时理解软件的整体结构和单个程序部分的细节.L. Merino 等[17]提出了一种交互式软件可视化工具 CityVR,该工具也在一个沉浸式的虚拟现实媒介中实现了城市隐喻;其粗粒度的代码可视化说明了软件工程任务的游戏化可以提高开发人员的参与度.R.Oberhauser 等[18]介绍了游戏化虚拟现实 FlyThruCode(GVR-FTC)技术,该技术将软件代码结构的沉浸式隐喻可视化游戏化,实现了宇宙和陆地 2 个隐喻;在陆地隐喻中,标有玻璃泡沫的城市代表包,建筑物代表类;在宇宙隐喻中,包由太阳系表示,类由行星表示.实验表明:VR 游戏化可以提高程序结构依赖性和代码模块化的理解力,但该工具依然属于类级别的代码可视化.

2　可视化可交互的虚拟城市

JavaCity 是一个以虚拟现实技术为媒介的互动的代码可视化工具,它通过非文本的视角描述项目的抽象结构,将软件项目中的各种度量可视化为不同的 VR 模型.开发人员可以在项目中自由探索、理解项目的代码结构及其复杂性.

2.1　JavaCity 概述

JavaCity 的设计目标是帮助 Java 工程师在不阅读代码的情况下更有趣、更直观地理解软件的整体结构及其复杂性.为了实现这个目标,采用城市隐喻技术和虚拟现实技术对软件项目可视化.城市隐喻依赖于软件结构和城市之间的相似性,并已经被证实是支持开发人员进行软件可理解任务的有效技术手段[19].这一方法也出现在现在的热门游戏中,

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

比如《模拟城市》《侠盗飞车》等.JavaCity 采用了 JavaScript 编写的 WebGL 第三方库 Three.js 来实现该目标.这是因为 Three.js 是完全开源的,且以更简单、更直观的方式封装了在 3D 图形编程中的对象.使用此技术开发的项目可以直接在浏览器上演示,相对于需要安装应用程序才能实施的其他技术,它具有方便快捷的优势.

为了使软件项目的度量信息映射到城市隐喻中,需要对软件项目的源代码进行解析;提取项目的软件标识符和其他的信息,其中软件标识符包括包、类、方法、参数以及代码行数量信息等.JavaCity 的设计原则是保持项目的原有层次结构不变.因此,还需要抽取项目的结构信息,即在项目中包、类、方法等信息的所属关系以及类之间的关系.根据具体可视化任务的需要提取其他的代码信息,如代码克隆片段和缺陷代码.详细的度量信息如表 1 所示.

表 1 度量信息表

度量	视觉特性	意义
包	社区	有助于理解包的结构等信息
类	小区	有助于理解类的结构等信息
方法	建筑物	有助于理解方法的结构等信息
形参	建筑物底面积	有助于识别参数列表过多的方法
代码行数量	建筑物高度	有助于理解复杂性
类的关系	管道	有助于理解软件的内聚和耦合强度

JavaCity 将整个软件项目表示为一座可交互的虚拟的城市,这种 3D 环境的导航方式有助于增强趣味性.图 1 显示了 JavaCity 的虚拟城市结构,将软件项目的包映射为在城市中的社区,类映射成小区,方法映射成建筑物.选择易于理解的源代码的度量信息表示建筑物的视觉变量,如建筑物的高度由所在方法的代码行数量决定,方法形参的数量反映建筑物的底面积.在软件项目中,类之间的关系使用管道连接,开发人员可以点击相应的类模型查询类与类之间的关系.在点击进入建筑物内部后,通过内墙面显示源代码(代码墙见图 2).

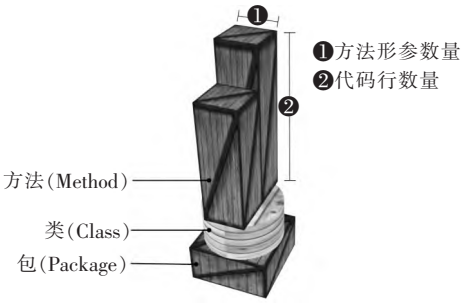


图 1 JavaCity 的虚拟城市结构

图 2 代码墙

2.2 JavaCity 实现要素

JavaCity 将软件项目的源代码作为输入数据,首先解析并提取在项目中的所有软件标识符,包括包、类、方法、属性等信息;然后针对不同的可视化任务提取内部的其他信息;紧接着将这些信息生成对应的 VR 隐喻模型;再根据可视化的任务选择模型的布局方式;最后开发人员就可以通过 JavaCity 工具执行指定的可视化任务.图 3 是 JavaCity 可视化工具的框架图.

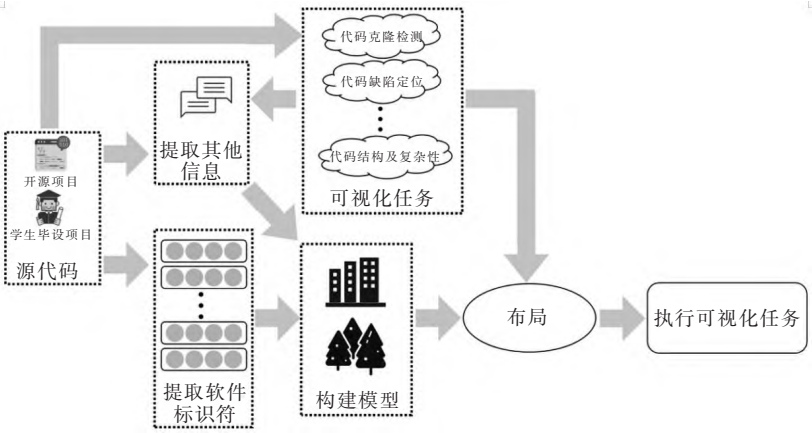


图 3 JavaCity 框架图

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

2.2.1　*提取软件标识符和其他信息*　JavaCity 将软件项目的源代码作为输入,解析提取在项目中的软件标识符和软件的其他的信息.标识符信息包括主要的面向对象元素,如包、类、方法、参数以及代码行数量信息等.JavaCity 在源代码上使用 JavaParser (https://javaparser.org/)解析器来构建抽象语法树(AST).AST 以树状的形式表现编程语言的结构,树的每个节点(AST Node)都表示在源码中的一个结构,AST 把在 Java 中的各种元素(如类、属性、方法、注释)定义成相应的对象.然后通过遍历整个 AST,提取出在项目中的所有代码元素,并将其父节点和子节点关系和类的调用关系保存在动态数组中.为了保存方法的代码行数量信息,工具使用在 Java 字节流中的 LineNumberReader 类进行处理.对于类的实现信息,工具仅提取项目创建的接口信息,不提取在项目中的第三方接口,首先利用 interface 关键词识别出在项目中创建的接口类,再利用 implements 关键词提取类的实现关系.对于类的继承关系,工具使用extends关键词提取项目创建的类的继承关系,忽略继承在第三方库中的类的继承关系.

2.2.2　*构建模型*　采用城市隐喻的技术构建目标隐喻模型,将源代码度量与虚拟城市的实体构建映射,这种映射是基于程序理解、软件可视化和开发获得的经验. JavaCity 使用 WebGL 的第三方库 Three.js 实现 3D 模型和虚拟环境的构建.

一个完整的虚拟现实系统主要包括场景(Scene)、相机(Camera)、光源(Light)、渲染器(Renderer)、控制器(Control)和物体(Mesh)6 个部分[20]. JavaCity 选择使用大海和天空的场景.由于使用透视投影相机获得的结果是类似人眼在真实世界中看到的有“近大远小” 的效果,更贴近真实世界,因此选择使用透视投影相机.而室外光源使用的是环境光(AmbientLight),室内使用聚光灯光源(SpotLight).对于渲染器,使用 WebGLRenderer 对象,通过在计算机上的显卡来渲染场景.控制器使用的是 TrackballControls 轨迹球控件.

对于在场景中的物体,将每种源代码度量可视化为对应的 VR 模型.JavaCity 使用在 Three.js 中不同的基础几何模型表示项目内源代码度量,包映射成在城市中的社区,使用 BoxGeometry 3 维立方体表示;类映射成小区,使用 CylinderGeometry 3 维圆柱表示;而在类中的方法映射成建筑物,也使用 CylinderGeometry 3 维立方体表示.利用建筑物的视觉变量表示方法的其他信息,选择易于理解的变量,建筑物的高度由所在方法的代码行数量决定,方法形参的数量反映在建筑物的底面积大小上;类之间的关系通过使用管道连接,使用 Shape 圆弧来表示;建筑物内部通过 6 面墙壁组成,使用 PlaneGeometry 平面几何体表示.

2.2.3　*布局*　最后一步是模拟真实城市布局,将所有源代码度量的 VR 模型在虚拟城市中布局.本文的布局方式受到了 CodeCity[4-5] 的启发,但由于 JavaCity 的可视化粒度变小了,因此本文的布局方式有所变化.JavaCity 根据项目文件结构的先后顺序依次排序,即先根据包在文件中的先后顺序从左到右排序,在包中的类和在类中的方法也如此.本文的设计原则要求可视化结果保持项目的层次结构,将类的模型层叠在包的模型构件的上面,方法的模型构件在类的模型构件上.仿佛虚拟城市一样,形成了一条条街道(见图 4).

通过使用这种布局,开发人员可以在不阅读源代码的情况下通过观察社区、小区和建筑物的形状就能识别出包、类、方法的模型构件大小,从而理解代码的结构和复杂性以及识别出与类和方法相关的代码(如大型类、大型方法、长参数列表),并且通过查看类的关系了解软件各模块间的内聚和耦合强度.

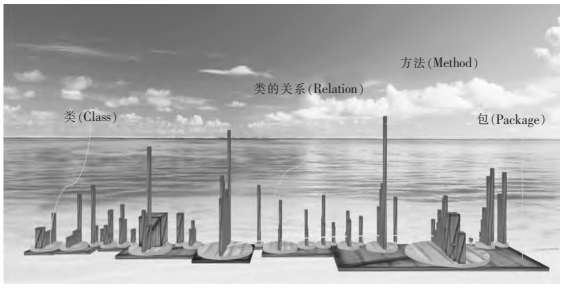


图 4　某项目代码可视化结果

3　实验评估

在软件开发过程中,通过有关实验,以验证基于虚拟现实技术和城市隐喻技术的代码可视化工具 JavaCity 在提高代码理解的时间效率、有用性和认知负荷上的效果.给定若干个与编程相关的任务,让不同经验的开发人员使用代码可视化工具 JavaCity 和传统的代码编辑器 IDEA 完成任务,比较他们在理解代码结构和代码复杂性方面究竟谁更占优势.即本实验将解决以下 3 个研究问题:

1)IDEA 和 JavaCity 哪一个在理解软件项目的

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

结构和复杂性上更能节省时间?

2)IDEA 和 JavaCity 哪一个能更全面理解软件项目的结构和复杂性?

3)IDEA 和 JavaCity 哪一个在理解软件项目的结构和复杂性上认知负荷更低?

3.1 数据集

评估实验的数据集选用代码在可视化工作[21-23]中常用的数据集 FindBugs(http://findbugs.sourceforge.net/)和 Azureus(https://sourceforge.net/projects/azureus/).FindBugs 是一个静态分析工具,能帮助检查 Java 项目隐藏的缺陷并提高代码质量.Azureus 是一款非常受欢迎、安全且值得信赖的 Torrent 客户端.表 2 显示了 2 个数据集的详细信息.

表 2 FindBugs 和 Azureus 数据集的特征

项目	类数量	方法数量	代码行	规模
FindBug	1 320	10 123	93 310	中型
Azureus	4 656	24 644	454 387	大型

3.2 参与者与设备

在实验评估时,由于需要参与者通过使用 JavaCity 工具和 IDEA 代码编辑器对 Java 源代码进行理解和分析,因此,要求被选中的参与者至少有半年的 Java 编程经验和熟练使用 IDEA 的经验.本研究招募了 32 名具有 Java 编程经验的参与者,包括 16 名高级软件工程师和 16 名初学者.

为了保证实验结果的准确性,实验要求所有的参与者统一使用同一套设备,按照顺序完成指定任务.

3.3 设计实验

针对本节最初提出的 3 个研究问题,设计了实验比较 JavaCity 和 IDEA.对于前 2 个问题,根据参与者完成任务的完成时间和正确性进行单独分析;最后一个评估使用 JavaCity 是否可以减轻认知负荷是一个主观问题,因此依据认知负荷理论对 NASA-TLX 工作负荷量表打分获得研究结果.

为了设计一个更加合理的实验,使用混合效应设计进行研究;将 32 名参与者根据编程经验分为初学者和高级软件工程师;每一类均有 2 组,总共 4 组,每组 8 个人.相同经验的 2 组参与者使用 IDEA 的平均时长是相同的,然后要求参与者先后使用不同的工具和不同的软件项目执行软件开发任务.在使用 IDEA 过程中不可以安装其他的插件,每个任务的时间限制 5 min,记录每个任务的完成时间.参与者的分组情况如表 3 所示.

表 3 参与者分组情况

组别	经验	第 1 次工具	第 1 次项目	第 2 次工具	第 2 次项目
A	初学者	IDEA	FindBugs	JavaCity	Azureus
B	初学者	JavaCity	FindBugs	IDEA	Azureus
C	高级软件工程师	IDEA	FindBugs	JavaCity	Azureus
D	高级软件工程师	JavaCity	FindBugs	IDEA	Azureus

对于问题 1)和问题 2)这 2 个研究问题,其研究目的是验证工具是否以更愉悦更直观的方式理解软件项目的代码结构和复杂性.因此,根据在可视化和代码理解中的常用任务[2,23],综合设计了 4 项开发任务,任务列表信息如表 4 所示.在表 4 中前 3 个任务是关于寻找在项目中的 Bad Smell 的任务,最后一个任务是关于类的关系的任务.T_1 任务是寻找在项目中的大型类(large class),大型类会使项目变得臃肿,很难理解,且不利于维护.若将其拆分成若干小类,则将可以避免代码和功能的重复.T_2 任务是寻找在项目中的长方法(long method),同样,长方法也很难理解,不利于代码复用,通过重构,从长方法中提取出多个新方法,在原长方法内替换为对新方法的调用,原长方法体的代码会更简洁.T_3 任务是寻找在项目中的长参数列表(long parameter list)的方法,长参数列表的方法会使方法更为复杂,难以使用,若将其简化,则可以增强代码的可读性.T_4 任务理解类与类之间的关系,对于理解面向对象具有重要的作用和意义.

表 4 任务列表

任务	内容
T_1	找到在整个项目中方法数目最多的 3 个大型类
T_2	找出在整个项目中代码行数最多的 3 个方法
T_3	找出在整个项目中方法形参最多的 3 个方法
T_4	识别出与 A 类有依赖关系的所有类

研究问题 3)是与认知负荷有关的问题[24].认知负荷是表示在处理具体任务时加在学习者认知系统上的负荷的多维结构,这个结构由反映任务与学习者特征之间交互的原因维度和反映心理负荷、心

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

理努力和绩效等可测性概念的评估维度所组成.在每名参与者完成每组任务之后,将采用 NASA 工作负荷量表(NASA-TLX)测量参与者的认知负荷[25].NASA-TLX 工作负荷量表是目前国外研究报道较多的心理负荷主观评价量表,多用于人体功效学领域的研究,如宇航员、飞行员、铁路司机、驾驶员等[26].NASA-TLX 工作负荷量表可以划分为 6 个子项,分别是脑力需求(Mental demand)、体力需求(Physical demand)、时间需求(Temporal demand)、业绩水平(Performance)、努力程度(Effort)、挫折程度(Frustration),NASA-TLX 工作负荷量表详细描述如表 5 所示.

不考虑在 NASA-TLX 工作负荷量表中的体力需求,这是因为完成这项任务是一个脑力劳动,与本实验关系较小.另外,时间需求和业绩水平与前 2 个问题的完成任务时间和正确性一致,也不考虑.在完成每组任务后,将要求参与者对脑力需求、努力程度和挫折程度进行打分,每项 10 分.

表 5　NASA-TLX 工作负荷量表

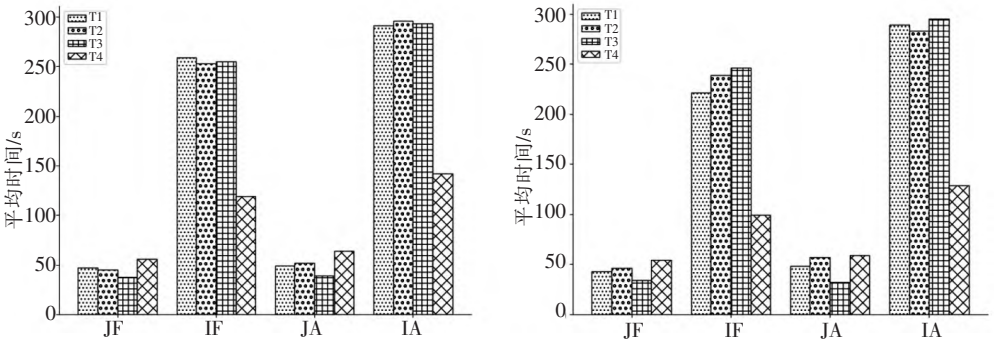
序号	子项	描述
1	脑力需求	付出了多少脑力劳动?使用起来简单还是复杂?
2	体力需求	付出了多少体力劳动?使用起来轻松还是费力?
3	时间需求	完成任务的进度匆忙或者紧张程度如何?
4	业绩水平	完成任务目标的满意度如何?
5	努力程度	付出了多少努力才能完成任务目标?
6	挫折程度	进行任务时感到不安、沮丧、急躁、烦恼的程度有多大?

3.4　实验结果与分析

本节将根据实验结果定性定量分析 JavaCity 和 IDEA 在理解软件项目的结构和复杂性方面的时间效率、有用性和认知负荷.设计的实验含有 3 个因素:经验、工具和数据集.经验因素分为初学者和高级软件工程师,工具因素分为 JavaCity 和 IDEA,数据集有 FindBugs 和 Azureus.

为了保证实验结果更加具有说服力,在实验过程中,记录了每名参与者完成每项任务的具体时间,在所有实验结束之后验证其正确性,最后再计算每一组 8 名参与者在该数据集上执行每项任务的平均时间和正确率.在参与者完成每组任务后,要求所有参与者对在使用当前工具过程中的脑力需求、努力程度和挫折程度进行打分.在正确性方面的验证工作中,成立了由 5 名具有丰富编程经验的开发人员组成的验证组对所有的结果进行验证.

首先,对于问题 1)的时间效率方面的研究,图 5 显示了所有参与者使用 JavaCity 和 IDEA 在 2 个数据集上完成任务的平均时间,横坐标表示使用不同工具在不同数据集上执行任务.观察图 5(a)和图 5(b),可以明显地看出参与者使用 JavaCity 比 IDEA 完成任务更节省时间,尽管所有的参与者在之前都有丰富的 IDEA 的使用经验,但是很多参与者仍在一些任务上甚至花费了整个 5 min 时间.这是因为参与者需要阅读和理解大量的代码才能完成任务.另外 JavaCity 与开发人员的编程经验无关,初学者和高级软件工程师使用相同工具在相同数据集上的平均时间基本相同,JavaCity 在处理大型软件 Azureus



(a)初学者完成任务的平均时间　　(b)高级软件工程师完成任务的平均时间

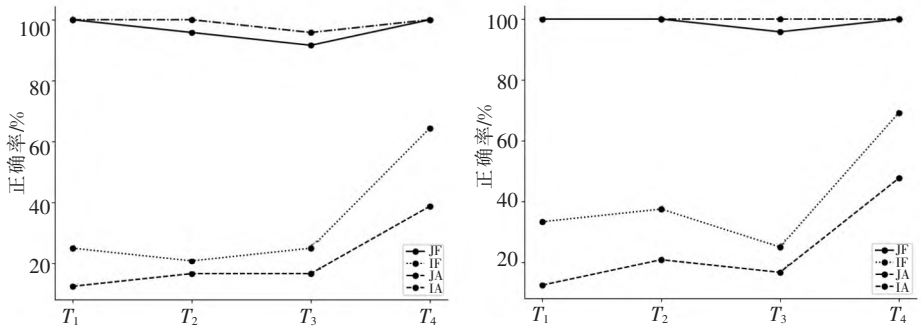
注:JF、IF、JA、IA 分别为 JavaCity-FindBugs、IDEA-FindBugs、JavaCity-Azureus、IDEA-Azureus.

图 5　参与者使用 JavaCity 和 IDEA 在 2 个数据集上完成任务的平均时间

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

和中型软件 FindBugs 上的平均时间也基本相同,这说明JavaCity具有很强的适应性.然而,IDEA 完成任务的平均时间与开发人员的编程经验有关,初学者 IDEA 在 FindBugs 执行 4 个任务上的平均时间分别为 259、253、255、119 s,而高级软件工程师的平均时间分别为 221、239、246、99 s,平均时间相差 20 s 左右.而且 IDEA 完成任务的平均时间还与软件规模有关,IDEA 在 FindBugs 和 Azureus 2 个数据集上的平均时间相差约 30 s.因此,可以回答问题 1),JavaCity 在理解软件项目的结构和复杂性时可以更节省时间.

对于问题2),图6 显示了所有参与者使用JavaCity和 IDEA 在 2 个数据集上完成任务的正确率.从图 2 可以清晰地看到初学者和高级软件工程师在使用 JavaCity 处理 FindBugs 和 Azureus 项目上执行 4 项任务的正确率都接近 100%.所有参与者使用 IDEA,它在 2 个项目上的正确率较低,只有在 T_4 任务上正确率超过了 50%.另外当 IDEA 处理大项目时,T_1、T_2、T_3 任务的正确率都明显下降了,即便是经验丰富的高级软件工程师也降到了 20%以下.出现这种现象的原因是:IDEA 在处理前 3 个任务时的阅读代码的工作量非常大,而且还需要记忆大量的信息;这对开发人员来说是一个巨大的挑战,而 JavaCity 在视觉上是非常直观的,阅读和记忆负荷轻.研究结果显示:JavaCity 在理解代码的结构和复杂性方面是非常全面的,而且与开发人员的编程经验和项目的规模相关性不明显.

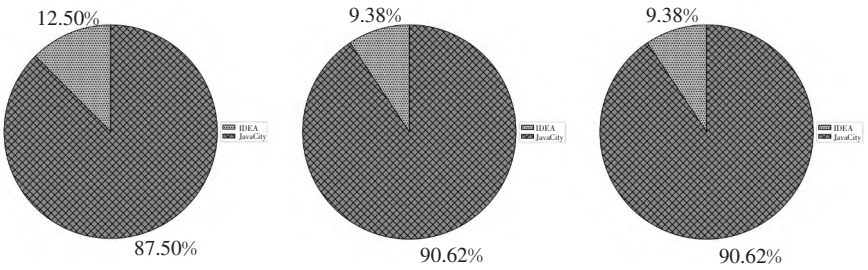


(a)初学者完成任务的正确率 (b)高级软件工程师完成任务的正确率

注:JF、IF、JA、IA 分别为 JavaCity-FindBugs、IDEA-FindBugs、JavaCity-Azureus、IDEA-Azureus.

图 6 参与者使用 JavaCity 和 IDEA 在 2 个数据集上完成任务的正确率

最后,对于问题 3),实验比较了每名参与者使用 2 个工具理解代码结构和复杂性的脑力需求、努力程度和挫折程度的打分,选出分数较低的工具.统计 2 个工具在较低的脑力需求、努力程度和挫折程度上的占比.图 7 显示了 JavaCity 和 IDEA 低认知负荷占比情况.结果表明:88%的参与者认为 Java-City 需要较低的脑力劳动,整个过程比较简单.91%的参与者认为 JavaCity 需要付出的努力程度较低.同样,91%的参与者认为在使用 JavaCity 过程中挫折很低,没有感到急躁和烦恼.12%的参与者认为IDEA需要较低的脑力劳动,9%的参与者认为 IDEA 需要较低的努力程度和较少的挫折.结果显示JavaCity在理解代码的结构和复杂性方面认知负荷更低.



(a)脑力劳动 (b)努力程度 (c)挫折程度

图 7 JavaCity 和 IDEA 低认知负荷占比情况

4 讨论

内部有效性威胁.在设计实验时,每名参与者先后使用不同的工具完成 2 次代码理解相关任务,参与者可能会疲倦.另外,由于在验证答案正确性方面,在文献中未找到与 2 个数据集关于这 4 项任务的正确答案,所以组成了一个经验丰富的验证组,收集所有参与者的答案,对所有的答案进行检查并

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

确定答案,这对本文给出的结论具有一定威胁性.服务器的配置可能会影响 JavaCity 的性能,但对实验结果影响较小.

外部有效性威胁.在实验评估时,选用了 FindBugs 和 Azureus 这 2 个项目.尽管在软件规模方面,FindBugs 属于中型软件,Azureus 是中大型软件,但是为了进一步证明 JavaCity 的通用性,有必要在更大的项目上进行实验.

构造有效性威胁.在评估 JavaCity 的性能方面,使用了时间效率、正确性和认知负荷 3 个指标;而对于认知负荷的测量,选择了 NASA 工作负荷量表(NASA-TLX).排除一些与实验关系较小的维度,最终选择了在 NASA 工作负荷量表中的脑力劳动、努力程度和挫折程度 3 个维度测量参与者的认知,本文基于这 3 个维度获得的结论的推广性还有待进一步讨论.

5　结论

本文提出了 JavaCity 工具,它以 VR 模型的形式呈现大型和复杂的软件项目的信息.该工具将软件项目可视化为一座虚拟城市,开发人员可以在城市中自由移动探索项目.招募的 32 名参与者评估了 JavaCity 和 IDEA 在执行一系列与用户理解软件项目相关的任务时的时间效率、有用性和认知负荷.实验结果表明 JavaCity 可以更加直观、更加有趣的方式理解大型软件项目的代码结构和复杂性,大大节省了项目理解的时间,从而降低软件开发的费用和提高软件的质量.

该工具具有强大的可扩展性,未来将在工具上扩展代码缺陷预测、代码克隆检测等功能展示在代码中更多复杂的信息,提升该工具辅助代码理解任务的能力.未来将提出方法级别的代码克隆检测模型,将其集成在该工具上,在代码墙上显示 4 种类型的克隆代码,对应的方法之间也可以通过天桥连接克隆对;将提出代码行级别的代码缺陷预测模型,丰富工具的功能,可以在代码墙上标注可能有缺陷的代码,对应方法在可视化工具上的颜色也可以根据方法蕴含缺陷率决定.

6　参考文献

[1] JOHNSON J, LUBO S, YEDLA N, et al. An empirical study assessing source code readability in comprehension [C]// MALETIC J, ROBINSON B. 2019 IEEE International Conference on Software Maintenance and Evolution (ICSME). Cleveland: IEEE, 2019: 513-523.

[2] KHALOO P, MAGHOUMI M, TARANTA E, et al. Code park: a new 3D code visualization tool [C]// ZHANG Kang. 2017 IEEE Working Conference on Software Visualization (VISSOFT). Shanghai: IEEE, 2017: 43-53.

[3] DOMINIC J, TUBRE B, HOUSE R J, et al. Program comprehension in virtual reality [C]//CHOI J. Proceedings of the 28th International Conference on Program Comprehension. New York: Association for Computing Machinery, 2020: 391-395.

[4] WETTEL R, LANZA M. CodeCity: 3D visualization of large-scale software[C]// WILHELM S. Companion of the 13th international conference on Software engineering-ICSE Companion'08. Leipzig: ACM Press, 2008: 921-922.

[5] WETTEL R, LANZA M. Visualizing software systems as Cities [C]//ANDRIAN M. 2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis. 2007: 92-99.

[6] ALANAZI R, GHARIBI G, LEE Y. Facilitating program comprehension with call graph multilevel hierarchical abstractions [J]. Journal of Systems and Software, 2021, 176: 110945.

[7] CHEERS H, LIN Y. Reverse engineering UML sequence diagrams for program comprehension Activities [C]//ELSTON C. 2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA). Sydney: IEEE, 2020: 1-10

[8] LI Zheng, WU Yonghao, PENG Bin, et al. SeCNN: a semantic CNN parser for code comment generation [J]. Journal of Systems and Software, 2021, 181: 111036.

[9] BAI Yang, ZHANG Liping, ZHAO Fengrong. A survey on research of code comment [C]//CHANG Chin-Chen, WANG Yulin. Proceedings of the 2019 3rd International Conference on Management Engineering, Software Engineering and Service Sciences. New York: Association for Computing Machinery, 2019: 45-51.

[10] ZHANG Chunyan, WANG Junchao, ZHOU Qinlei, et al. A survey of automatic source code summarization [J]. Symmetry, 2022, 14(3): 471.

[11] JIANG S. Improving program comprehension using neural machine translation [D]. South Bend: University of Notre Dame, 2018.

[12] KADAR R, OTHMAN J, WAHAB N A. A survey on ontology-based visualization techniques towards program comprehension application [J]. Journal of Computing Research and Innovation, 2018, 3(1): 19-29.

[13] ALNABHAN M, HAMMOURI A, HAMMAD M, et al. 2D visualization for object-oriented software systems [C]//

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

SABRI M A, YAHYAOUY A, TAIRI H et al. 2018 International Conference on Intelligent Systems and Computer Vision (ISCV). Fez: IEEE, 2018: 1-6.

[14] WETTEL R, LANZA M. Program comprehension through software habitability [C]//WIBG G. 15th IEEE International Conference on Program Comprehension (ICPC'07). Banff: IEEE, 2007: 231-240.

[15] ERRA U, SCANNIELLO G, CAPECE N. Visualizing the evolution of software systems using the forest metaphor [C]//CROLL P R. 2012 16th International Conference on Information Visualisation. Montpellier: IEEE, 2012: 87-92.

[16] HORI A, KAWAKAMI M, ICHII M. CodeHouse: VR code visualization tool [C]//FABRY J. 2019 Working Conference on Software Visualization (VISSOFT). Cleveland: IEEE, 2019: 83-87.

[17] MERINO L, GHAFARI M, ANSLOW C, et al. CityVR: gameful software visualization [C]//ZHANG L, ZIMMERMANN T. 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME). Shanghai: IEEE, 2017: 633-637.

[18] OBERHAUSER R, LECON C. Gamified virtual reality for program code structure comprehension [J]. International Journal of Virtual Reality, 2017, 17(2): 79-88.

[19] WETTEL R, LANZA M, ROBBES R. Software systems as cities: a controlled experiment [C]//TAYLOR R N. Proceedings of the 33rd International Conference on Software Engineering. New York: Association for Computing Machinery, 2011: 551-560.

[20] 王芳芳.基于 Threejs 技术的虚拟校园设计与实现[D].杭州:浙江工商大学,2017.

[21] ROMANO S, CAPECE N, ERRA U, et al. On the use of virtual reality in software visualization: the case of the city metaphor[J]. Information and Software Technology, 2019, 114: 92-106.

[22] MERINO L, BERGEL A, NIERSTRASZ O. Overcoming issues of 3D software visualization through immersive augmented reality [C]// ITURBIDE J A V. 2018 IEEE Working Conference on Software Visualization (VISSOFT). Madrid: IEEE, 2018: 54-64.

[23] MERINO L, GHAFARI M, ANSLOW C, et al. A systematic literature review of software visualization evaluation [J]. Journal of Systems and Software, 2018, 144: 165-180.

[24] 付倩文.基于认知负荷的 VR 任务情景设计研究 [D].贵阳:贵州大学,2021.

[25] QAYUM A, KHAN S U R, INAYAT-UR-REHMAN, et al. FineCodeAnalyzer: multi-perspective source code analysis support for software developer through fine-granular level interactive code visualization [J]. IEEE Access, 2022, 10: 20496-20513.

[26] 梁丽玲,赵丽,邓娟,等.NASA-TLX 量表的汉化及信效度检验 [J].护理研究,2019,33(5):734-737.

JavaCity: A Code Visualization Tool to Enhance Code Understanding

XIAO Yingjian[1,2], JIE Anquan[1*], LI Hongwei[1*], ZHONG Chongwen[1], LUO Rong[3]

(1.School of Computer and Information Engineering, Jiangxi Normal University, Nanchang Jiangxi 330022, China;
2.School of Computer and Information Engineering, Nanchang Institute of Technology, Nanchang Jiangxi 330044, China;
3.School of Digital Industry, Jiangxi Normal University, Shangrao Jiangxi 334000, China)

Abstract: In this paper, the immersive code visualization tool JavaCity is proposed based on the city metaphor, which enables fine-grained visualization at the method level and further enhances code visualization and code understanding by extracting various metrics of a software project to build a visual and interactive virtual city. Comparative experimental studies conducted on JavaCity and IDEA show that JavaCity saves more than 200 seconds in time efficiency on tasks related to understanding software projects, is nearly 100% correct on usefulness ratings, and nearly 90% of participants perceive lower cognitive load in terms of mental effort, effort level, and frustration level.

Key words: code visualization; urban metaphor; code understanding; virtual reality

(责任编辑:冉小晓)

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net