文章编号：2095-2163(2019)03-0085-04 中图分类号：TP311.5 文献标志码：A

基于变更块的代码重构模式展示—以抽取方法为例

石 伟，杨春花
（齐鲁工业大学（山东省科学院）计算机科学与技术学院，济南 250353）

摘 要：代码变更的理解一般基于文本差异化工具，其处理输出的基本对象是一个变更块（hunk）。许多重构都掺杂在日常的 bug 修改、功能特征的增加等代码变更中，将重构与其它工作隔离有利于对代码变更的理解。现有的代码可视化方法仅展示变更文本，未进行代码变更前后重构模式的展示，没有发挥出可视化技术的优越性。本文基于变更块对代码重构模式进行展示，以抽取方法为例，在 4 个开源项目中进行实验，并取得了理想的展示效果。

关键词：变更块；重构；抽取方法；重构模式展示

Code refactoring pattern display based on hunks—take the extract method as an example

SHI Wei, YANG Chunhua
(School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China)

【Abstract】 The understanding of code changes is generally based on text differentiation tools, whose basic object for processing output is a hunk. Many refactoring are incorporated into code changes such as daily bug fixes and functional features. Separating refactoring from other work facilitates understanding of code changes. The text differentiation tool only displays the changed text, and does not show the refactoring pattern before and after the code change, and does not play the superiority of the visualization technology. In this paper, code refactoring pattern is demonstrated based on hunks. Taking the extract method as an example, the experiment is carried out in four open source projects, and the ideal display effect is achieved.

【Key words】 hunk; refactoring; extract method; refactoring pattern display

0 引 言

代码变更的理解一般基于文本差异化工具，并且各种版本的管理工具中都集成了某种文本差异化工具，帮助用户查看文本代码变更情况，而文本差异化工具处理输出的基本单位就是变更块（hunk）。因此，基于变更块对代码重构模式进行分析展示，有利于将来可以较容易地集成到版本管理工具中。

重构[1]是调整软件结构的一种手段，在不对软件功能特征进行更改的条件下，提高软件的可理解性和可维护性。重构在软件演化过程中存在普遍性，检测并移除复杂代码已经成为软件生命周期中重构阶段的基本工作。在理解代码变更时，将变更数据抽取出来进行相应的可视化展示，将有助于代码重构分析人员区分哪些代码存在重构关系，帮助其高效地理解代码是如何进行变更的。目前，对于重构检测及可视化技术方面的研究，包括函数抽取重构的检测算法[2]、基于深度学习的代码克隆检测技术[3]、基于软件度量的函数提取重构方法[4]、基于索引的分布式代码克隆检测[5]、克隆代码可视化的研究[6]等。

本文设计了一种基于变更块的代码重构模式展示方法，将存在重构关系的变更块分组，并以恰当的方式展示，展示变更块类内、类间重构关系，以便于用户对重构数据进行分析和研究。

1 基于 hunk 的代码重构模式展示方法

展示方法针对一个源代码文件的 2 个连续修改版本进行分析处理，利用文本差异化工具对这 2 个连续版本进行变更块数据信息的抽取，将变更块数据进行相应的展示。然后，通过重构识别分析工具对变更块数据信息做出抽取、研究分析，判断其是否存在重构模式、存在哪种模式的重构模式。最后，利用重构模式识别过程获取的重构模式、变更内容等参数，结合不同的重构模式使用不同展示方式选取对应的重构模式展示模板，对重构模式给出相应的展示。展示方法框架如图 1 所示。展示方法的流程步骤描述见如下。

作者简介：石 伟（1990-），男，硕士研究生，主要研究方向：软件应用技术；杨春花（1974-），女，博士，教授，主要研究方向：代码变更分析、软件演化、软件开发。

收稿日期：2019-02-27

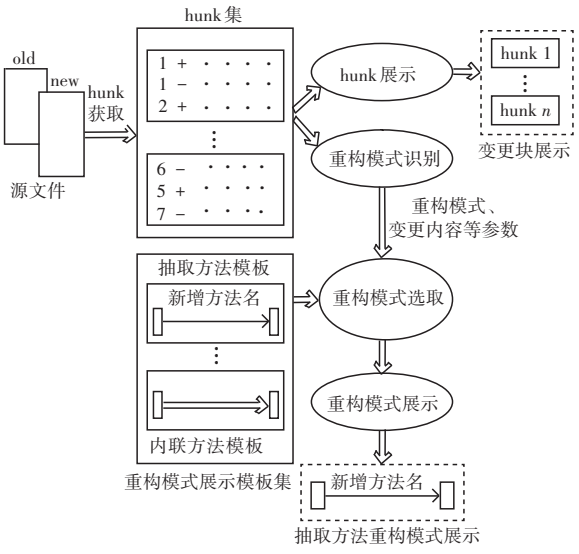

图 1 方法框架图

Fig. 1 Method framework diagram

(1)hunk 数据的获取。hunk 是一个代码变更块,可通过文本差异化工具进行获取,是文本差异化工具处理输出的基本单位。其中包含了代码变更前后删除或增加的代码信息及其行号范围信息,如图 2 和图 3 所示,都是一个 hunk。

(2)hunk 展示。hunk 展示方面普遍采用的方式有 2 种。一种是集中式展示,如图 2 所示。另一种是并排式展示,如图 3 所示。

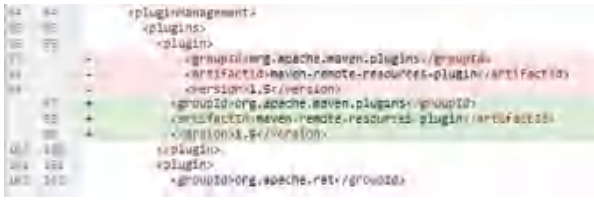

图 2 集中式展示

Fig. 2 Centralized display

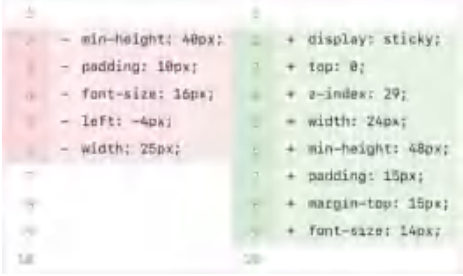

图 3 并排式展示

Fig. 3 Side-by-side display

(3)重构模式识别。利用重构模式识别算法对获取的代码变更块数据进行重构模式识别,判断其存在的重构模式,并获取重构模式名称、变更类型、变更实体、变更内容、行号范围等参数信息。

(4)重构模式展示模板集。由于重构模式的种类较多,不同的重构模式具有不同的特征属性,所以要对变更块数据存在的重构模式进行展示就不能采用单一的展示模式,而是要根据各种重构模式不同的特征属性,对不同的重构模式设计不同的展示模式。

(5)重构模式的选取。根据重构模式识别获取的重构模式参数,对重构模式展示模板集中的重构模式模板进行选取。

(6)重构模式展示。界面布局采用 JQuery EasyUI 布局插件 Layout 布局,通过嵌套进行复杂的界面布局,通过布局将界面分为相应的区域,分别展示不同的信息。

2 展示方法的实现—以抽取方法为例

2.1 方法实现

2.1.1 变更块获取与展示

变更块数据的获取采用 GNU Diff(http://www.gnu.org/software/diffutils/)文本差异化工具,其处理输出的基本对象是一个变更块(hunk)。GNU Diff 文本差异化工具可以显示文件是否不同,提供了一些方法来抑制某些看起来不重要的差异。最常见的是,这种差异是指文字或线条之间的空白量的变化。工具还提供了方法来抑制字母表中的差异,或者与匹配的正规表达式的行相匹配,将 2 个文档的内容参照比较,以便用户对文档内容信息进行分析和处理。

变更块数据的展示采用普遍使用的集中式展示,将同一个源文件的 2 个连续版本中的变更块数据信息进行合并处理,变更块前后变更数据代码行用不同颜色标注出来,每个变更块变更数据上下都保留 3 行未发生变更的代码行数据,并在变更块数据代码行前标明变更前后的代码行号,以便后续用于对变更块数据的理解、分析和处理。

2.1.2 重构模式分析处理与展示

重构模式的分析处理借鉴孙美荣等人[7]研究提出的识别变更代码重构模式算法,并引入一些改进对重构模式进行识别。原算法利用 ChangeDistiller(https://bitbucket.org/sealuzh/tools-changedistiller/src/)工具获取变更类型,JDiff(http://javadiff.sourceforge.net/)文本差异化工具进行变更块抽取,Levenshtein(https://en.wikipedia.org/wiki/Levenshtein_distance)算法进行字符串差异度量。其中,ChangeDistiller 工具是 Fluri 等人[8-9]编写的一个 Tree differ 算法。由于本文使用的文本差

异化工具是 GNU Diff 工具，即需将该算法中的文本差异化工具 JDiff 工具换成 GNU Diff 工具，以达到文本比较、变更数据获取的目的。重构模式分析处理的基本过程是：先根据 GNU Diff 文本差异化工具获取相应的代码变更块数据，再通过 ChangeDistiller 工具获取代码变更的所有类型，记录其行号范围，同时将根据相关的父类实体对新增语句和删除语句进行分组，获得一个元组集合。根据 GNU Diff 文本差异化工具获取相应的代码变更块，将获取的新增代码变更块和删除代码变更块进行相似度比较，判断是否存在代码变更块抽取。最后，对存在变更块抽取的文件进行重构模式判断，判断其是否符合相应的重构模式关系，并对相关重构模式变更数据进行抽取。

重构模式的展示根据重构模式识别获取的重构模式参数选取相应的重构模式模板进行重构模式展示。

2.2　方法验证

数据来源于通过 MiniGit（https://github.com/Software Introspection Lab/MiniGit）工具获取的 4 个开源项目数据，以验证展示方法实现的可行性。4 个开源项目分别为：eclipse.jdt.core、google－guice、jEdit、maven，各项目信息参见表 1。

表 1　开源项目信息

Tab. 1　Open source project information

项目名称	时间范围	总版本数
eclipse.jdt.core	2001-08-23~2013-12-21	19 352
google-guice	2006-10-15~2013-12-25	1 205
jEdit	1998-10-17~2012-08-18	6 189
maven	2003-10-14~2014-02-26	9 735

本文以抽取方法重构模式为例，对存在重构模式的代码变更块数据进行展示。抽取方法重构模式就是将一个过长的方法或一段需要注释用途的代码放入一个独立的方法中，用独立方法的名称解释该方法的用途。此示例的源文件数据取自 maven 开源项目，2 个连续的文件提交版本分别是：684eed4a0c97bbbcc6e01432105d5b15e0e4f9b2 和 a48ae318c671c99048010183227d078d19dc972b，文件名称是 DefaultMavenExecutionRequestPopulator.java。首先，通过 GNU Diff 文本差异化工具对 2 个文件进行变更块数据的获取，然后通过重构识别工具对变更块数据开展重构模式识别，获取变更语句的详细信息，包括变更类型、变更实体、变更内容、行号等，对存在重构模式的变更块数据进行提取，选择抽取方法重构模式的相应展示模式实现效果展示。

抽取方法重构模式展示有 3 个参数，分别为：被抽取变更块、抽取变更块、新增方法名。其中，被抽取变更块有时不止有一个，是可增加的。如图 4 所示。

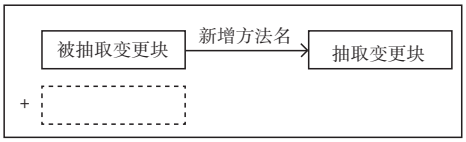

图 4　展示参数图

Fig. 4　Display parameter diagram

此示例源文件的 2 个连续文件提交版本数据的获取是利用 XLoadTree 组件对数据库数据按数据的仓库信息进行分层，展示每个仓库中的所有文件修改版本，再对每个文件修改版本下的所有文件进行分层展示得到的。XLoadTree 组件是基于 AJAX 和 XML 的动态加载 JS 树组件，通过对树节点的 source 属性定义指向一个 xml 文件，从而使数据进行载入，还可以包含嵌套的子节点以及指向其它 xml 文件的子节点，利用 DOM 进行转换。本地数据目录树如图 5 所示。

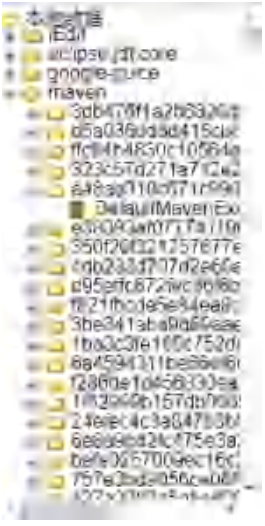

图 5　本地数据目录树

Fig. 5　Local data directory tree

变更块数据展示采用集中式展示的方式，将每个变更块的数据信息展示到界面上，变更块展示如图 6 所示。

重构模式展示通过对变更块数据的分析处理，获取存在抽取方法重构模式的变更块，本示例中被抽取变更块有 2 个，将重构模式中被抽取变更块数据填充到左边框架，抽取变更块数据填充到右边框架，中间用箭头和新增方法名解释该方法的用途，以便对重构模式的分析和理解。重构模式展示如图 7 所示。

图 6 变更块展示

Fig. 6 Hunk display

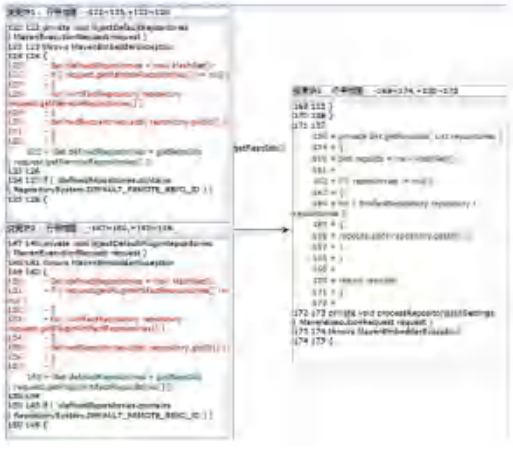

图 7 重构模式展示

Fig. 7 Refactoring pattern display

3 结束语

本文设计了一种基于变更块的代码重构模式的展示方法,以抽取方法为例实现了对一个源文件的2个连续版本中存在抽取方法重构模式的相关展示,使代码重构变得更加简单、直观,发挥出了可视化技术的优越性,为理解代码重构提供了便利。

参考文献

[1] FOWLER M, BECK K, BRANT J, et al. Refactoring: Improving the design of existing code[M]. Sebastopol, CA: Addison-Wesley Professional, 1999.

[2] 刘阳,刘秋荣,刘辉. 函数抽取重构的自动检测方法[J]. 计算机科学,2015,42(12):105-107.

[3] 刘复星,魏金津,任女尔. 基于深度学习的代码克隆检测技术研究[J]. 电脑知识与技术,2018,14(18):178-179,185.

[4] 冯燕,肖笛. 基于软件度量的函数提取重构初探[J]. 数字技术与应用,2017(6):254,256.

[5] 林婵,李俊杰,饶飞,等. 基于索引的分布式代码克隆检测[J]. 信息安全研究,2016,2(3):201-210.

[6] 何蔷. 克隆代码可视化系统的设计与实现[D]. 哈尔滨:哈尔滨工业大学,2015.

[7] 孙美荣,杨春花. 基于变更类型和相似性比较的代码重构模式识别[J]. 智能计算机与应用,2018,8(2):25-29,34.

[8] FLURI B, GALL H C. Classifying change types for qualifying change couplings [C]//Proceedings of the 14th International Conference on Program Comprehension. Athens, Greece: IEEE, 2006:35-45.

[9] FLURI B, WUERSCH M, PINZGER M, et al. Change distilling: Tree differencing for fine-grained source code change extraction [J]. IEEE Transactions on Software Engineering, 2007, 33(11): 725-743.

(上接第 84 页)

[5] JIAN Bing, VEMURI B C. Robust point set registration using gaussian mixture models [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 33(8): 1633-1645.

[6] TAO Wenbing, SUN Kun. Asymmetrical gauss mixture models for point sets matching [C]. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR). Columbus, Ohio, USA: IEEE, 2014: 1598-1605.

[7] MYRONENKO A, SONG Xubo. Point set registration: Coherent point drift [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32(12): 2262-2275.

[8] 秦红星, 徐雷. 基于信息论的 KL-Reg 点云配准算法[J]. 电子与信息学报, 2015, 37(6): 1520-1524.

[9] 王安娜, 吕丹, 王哲, 等. 基于 SIFT 特征提取的非刚性医学图像配准算法研究[J]. 生物医学工程学杂志, 2010, 27(4): 763-768,784.