开源环境下开发人员行为特征挖掘与分析

袁 霖[1),2)] 王怀民[1)] 尹 刚[1)] 史殿习[1)] 李 翔[1)]

[1)](国防科学技术大学计算机学院 长沙 410073)

[2)](信息工程大学电子技术学院 郑州 450004)

摘 要 软件项目开发人员行为特征是软件工程领域所关注的重要问题之一,获取个体行为特征可用于评估项目发展的进度、认识项目的发展特征、发现制约项目发展的瓶颈以及发觉项目开发过程中的异常现象.文中基于 Alitheia 平台设计并实现了两个测度插件,并结合一些著名开源软件项目的版本控制系统 SVN 库中的相关数据,对参与项目的开发人员的多种贡献行为和协同行为进行度量和深入分析,发现了一些具有重要理论和实践价值的现象与结论,揭示了开源模式下开发人员的部分行为特征.

关键词 软件资源库;数据挖掘;行为特征;贡献;协同

中图法分类号 TP393 **DOI 号**: 10.3724/SP.J.1016.2010.01909

Mining and Analyzing Behavioral Characteristic of Developers in Open Source Software

YUAN Lin[1),2)] WANG Huai-Min[1)] YIN Gang[1)] SHI Dian-Xi[1)] LI Xiang[1)]

[1)](*School of Computer, National University of Defense Technology, Changsha* 410073)

[2)](*School of Electronic Technology, Information Engineering University, Zhengzhou* 450004)

Abstract The behavioral characteristic of developers in software projects is always an important problem in software engineering and can be performed to monitor the rate of project development, identify implementation bottlenecks, isolate exceptional cases, and help with future project planning. Analyzing and evaluating software development process and source code characteristics is an important step towards achieving behavioral characteristic. The Alitheia is a platform modeled around a pluggable, extensible architecture that enables it to incorporate various types of data sources and be accessible through various user interfaces. In this research, two metric plug-ins based on Alitheia platform were designed and implemented, which are called developers contribution behavioral metric plug-in and committer network construction plug-in. With the Subvision repositories which were collected from several famous open source software projects in the world, some kinds of contribution and collaboration behavior were measured by these two plug-ins and analyzed in detail. And then, several important phenomena and conclusions which are both theoretical and practical were discovered, and part of behavioral characteristic of developers in open source development environment is presented in this paper. This work prepares the ground for quantification and utilization of the software trustworthiness evidence from the data about developers of open source software and will help evaluate trustworthiness of open source software efficiently and automatically.

Keywords software repository; data mining; behavioral characteristic; contribution; collaboration

收稿日期:2010-08-22.本课题得到国家"八六三"高技术研究发展计划重点课题(2007AA010301)、国家自然科学基金(60903043)和核高基重大专项课题(2009ZX01043-001)资助.**袁 霖**,男,1981 年生,博士研究生,讲师,主要研究方向为软件可信评估和数据挖掘. E-mail: fkefss@gmail.com. **王怀民**,男,1962 年生,博士,教授,博士生导师,主要研究领域为分布式计算、可信软件、网络与信息安全. **尹 刚**,男,1975 年生,博士,讲师,研究方向为可信软件、分布计算与信息安全. **史殿习**,男,1966 年生,博士,副教授,研究方向为分布计算与自主计算. **李 翔**,男,1988 年生,硕士研究生,研究方向为数据挖掘技术.

1 引 言

软件生命周期内项目参与人员的行为特征是软件工程领域所关注的重要问题.获取个体行为特征可用于评估项目发展的进度、认识项目的发展特征、发现制约项目发展的瓶颈以及发觉开发中的异常现象.软件项目参与人员的行为是多方面的,既包括源代码的编写与调试,也包括参与项目小组讨论、进行项目规划、设计等,但是并非项目参与人员所有行为都可以获取并合理量化,因此对项目参与人员各种行为的分析一直是困扰软件工程领域的问题之一.而随着现代开发模式的转变,特别是开源运动的兴起以及各种项目管理工具的广泛应用,软件开发过程中各种行为相关的数据得以规范和记录,为该问题的研究提供了新的思路与机遇.

在开源环境下,项目参与人员以各种行为方式参与到项目开发当中.其中,既包括源代码的编写、变更与调试,也包括缺陷的报告、修订以及对项目下一步发展的讨论与规划等.通过各种项目管理工具,这些信息被完整的记录在各种软件资源库中,其中包括版本控制系统(如 SVN 和 CVS 库)、缺陷跟踪系统(如 Bugzilla 和 Tracker 库)以及通信管理工具(如邮件系统和 IRC)等.这些软件资源库中包含了丰富的、各种类型的关于开发人员和用户行为的记录,为进一步的研究工作提供了一个良好的、可跟踪的数据资源平台,使得通过数据挖掘的方式获取项目参与人员的各种行为特征成为可能.

2 相关工作

当前,在软件资源库挖掘(Mining Software Repository,MSR)领域,与开发人员行为相关并受到广泛关注的研究方向主要有两个:(1)对参与项目的开发人员贡献行为的度量方法的研究;(2)在开源开发模式下对开发人员协同方式的分析.

2.1 人员贡献行为度量

在软件项目中,如何对开发人员的贡献度进行度量一直是研究人员所关注的问题,也是分析开发人员行为特征的重要基础.

在传统软件开发模式中,最初的开发人员对项目贡献的度量方法是通过工作单元中的单词量[1](words);随着软件程序设计语言的发展,开发人员对项目贡献的度量转变为程序员编写的代码行数(Lines of Code,LoC);进一步地,研究人员提出了多种度量单位如功能点(function points)、面向对象中的类、函数以及构件等度量单位.上述传统度量方法中,主要是面向项目最终产品(即源代码)的度量,也就是说在项目结束之前无法准确获取度量结果.然而,随着近年来软件开发方式的发展,当今的软件开发人员不仅要开发代码,还需要与相关人员进行频繁的交互与协同以及使用各种开发辅助工具,这种趋势随着开源运动的兴起愈加明显.从这个角度来说,开发人员的贡献行为应该从多方面来进行综合的度量,仅以代码行等作为有效测度的度量方法显然不能满足这种需要.

出于上述考虑,Gousios[2]等人总结了开源环境下项目参与人员参与项目的各种行为方式(见表 1),其中类型列中,P 表示该行为对项目具有正面影响,可视为积极行为;N 表示对项目有不好的影响,可视为消极行为.进而在此基础上,Gousios[2]等人设计了一种综合的人员贡献度计算方法,见式(1)和(2).

表 1 开源开发人员贡献行为方式

资源库	行为	标识	类型
代码及文档库	添加代码行	CADD	P
	删除代码行	CREM	P
	更改代码行	CCGN	P
	提交新文件	CNS	P
	提交新目录	CND	P
	提交的代码产生一个 Bug	CGB	N
	提交代码解决一个 Bug	CCB	P
	添加/变更代码文档	CAD	P
	修改代码类型	CSF	P
	一次提交动作中提交超过多个文件	CMF	N
	提交文档文件	CDF	P
	提交翻译文件	CTF	P
	提交二进制文件	CBF	N
	空提交行为	CEC	N
	提交注释包含缺陷号	CBN	P
邮件库	第一个相应一个会话	MFR	P
	开始一个新会话	MST	P
	恶意攻击会话	MFW	N
	关闭长期的会话	MCT	P
缺陷库	关闭已解决缺陷	BCL	P
	报告缺陷	BRP	P
	关闭尚未解决的缺陷	BCR	N
	注释缺陷	BCC	P
Wiki	开启新页面	WSP	P
	升级页面	WUP	P
	关联页面与文件	WLP	P
IRC	频繁参与 IRC 讨论	IFP	P
	直接回答问题	IRQ	P

$$C_i(d_x)=\frac{c_i(d_x)}{\sum_{j=1}^{k} c_i(d_j)} \quad (1)$$

$$C_{\text{tot}}(d_x)=\sum_{i=1}^{n} w_i C_i(d_x) \quad (2)$$

公式中各符号含义如下：

k：项目中开发人员总数；

n：贡献行为类型总数；

$c_i(d_x)$：开发人员 d_x 在第 i 类行为上的贡献量；

$C_i(d_x)$：开发人员 d_x 在第 i 类行为上的贡献度，即 d_x 的该类行为所占该行为总量的比例；

w_i：第 i 类行为的权重，其中，如果是积极行为 $w_i>0$，反之 $w_i<0$，且 $\sum_{i=1}^{n}|w_i|=1$；

$C_{\text{tot}}(d_x)$：开发人员 d_x 的总体贡献度.

由于不同行为对项目的影响程度(权重 w_i)不仅与项目进展的具体阶段有关，甚至还与评价者的主观感受密不可分，因此不同类型的贡献行为(如源代码提交和漏洞报告)对项目的影响程度大小很难判断，权重 w_i 的设定成为一个难点. 而由于目前尚缺乏得到广泛认可的权重设定的理论依据，因此首先考虑在软件资源库中通过数据挖掘的方法自动化地计算 $C_i(d_x)$，而 $C_{\text{tot}}(d_x)$ 的具体计算方法需要由研究人员、评估人员或项目管理者根据具体情况和具体领域知识来自行决定.

2.2　协同开发行为分析

与传统软件开发不同，开源软件开发过程中没有严格的过程模型和计划管控. 在开源环境下，开发人员可能互相从未谋面，但却能协同开发出像 Linux、Apache 等许多获得巨大成功的优秀软件. 开源环境下，软件项目的开发过程如何进行组织管理以及开发人员的协同开发行为应当具有什么样的特点等问题引起了越来越多的研究人员的关注. 其中代表性的工作是 Lopez-Fernandez[3] 等人以版本控制系统 CVS 日志数据为挖掘对象，分析并构造了反映协同开发行为的提交者网络(commiter network). 其定义的提交者网络是一个无向图，顶点代表开发人员，边代表开发人员之间的协同关系. 如果一对开发人员所参与(修改、删除或添加等行为)的模块集合存在交集，则其所在定点之间存在一条边，边的权重为交集中模块个数的多少，这个值被称为开发人员之间的协同次数. 台湾研究人员 Huang[4] 等也采取了类似的方法构造了开发者网络，并在此基础上分析项目参与人员对项目的影响程度，进而识别项目开发过程中的核心开发人员与外围开发人员. 在本课题组之前的研究中，对参与项目的开发人员的角色信息进行挖掘和分析，给出了角色与项目排名之间的关联关系[5-6].

3　基于 Alitheia 的测度插件的设计与实现

3.1　Alithia 平台介绍

制约当前 MSR 研究工作的一个重要因素是缺乏统一的元数据分析平台，所有的 MSR 工作都需要首先对软件资源库中的数据进行解析和存储，这使得研究人员需要分别开发自己的元数据分析工具，这是一项重复而又繁琐的工作，极大地浪费了相关研究人员的精力. 并且由于各种数据分析工具相互之间缺乏统一的规范与数据格式，给 MSR 工作的进一步深入造成了严重的阻碍. Alitheia 平台是针对此需求而开发的、可用于软件工程研究的可扩展平台[7]，其目标是为研究人员提供自动化的数据分析及相关公共服务，使研究人员可以把精力放在问题研究本身. 该平台还提供了接口化的软件开发测度插件服务，方便用户根据特定工作需要自行进行扩展.

目前 Alitheia 还处于测试、开发阶段，没有推出正式发行版，有很多功能并不完善. 平台支持的资源库只有软件控制管理系统 SVN、漏洞跟踪系统 Bugzilla 以及邮件列表，而 GIT 软件控制管理系统、IRC 交流信息等数据尚不能处理.

3.2　测度插件

Alitheia 平台可通过开发测度插件来扩展，测度插件是实现了相同接口的 OSGi 服务. 一个测度插件可以定义一个或多个测度，一个测度可以有一个或多个触发类型(activation type). 测度的计算过程可以利用元数据，也可以依赖于其它的测度结果. 测度触发服务(metric activator service)使用测度之间的依赖信息来决定测度的执行顺序. Alitheia 0.95 自带的测度插件描述如表 2 所示.

3.3　贡献行为测度插件

贡献行为测度插件[8] 的目标是从软件资源库中解析出项目参与人员的各种贡献行为，进而计算人员对项目的贡献度. 表 3 描述了 Alitheia 实现的贡献行为测度插件可分析的各种行为类型及其性质.

表 2 Alitheia 平台提供的测度插件

测度插件	描述	触发类型	测度数量	依赖关系
项目规模	计算项目规模有关的各种测度，比如源文件数目、文档数目、源代码行数、注释行数、文档行数等	ProjectFile ProjectVersion	11	无
模块规模	将项目规模测度的结果按照源代码目录的粒度聚集为模块，并对每个版本计算源代码平均模块大小	ProjectVersion ProjectDirectory	3	项目规模
代码结构	计算 c/c++和 java 源代码的 McCabe 与 Halstead 测度，以体现代码结构特征	ProjectFile	15	无
贡献行为	从代码库、漏洞数据库、邮件列表中分析出各种开发人员的贡献行为	MailingListThread ProjectVersion Bug	1	项目规模
测试用例	计算一般单元测试框架下的测试用例个数	ProjectFile	1	无
可维护性	计算每个源代码模块的可维护性指数，最后对整个软件项目求平均值	ProjectDirectory ProjectVersion	2	项目规模 模块规模 代码结构

表 3 贡献行为 SDM 插件支持的贡献行为类型

贡献行为类型	英文全称	性质
CNS	Commit new source file	P
CND	Commit new directory	P
CDF	Commit documentation files	P
CTF	Commit translation files	P
CBF	Commit binary files	N
CEC	Commit with empty commit message	N
CMF	Commit more than X files in a single commit	N
CCNG	Total lines modified until current version	P
CADD	Total lines added until current version	P
CREM	Total lines removed until current version	P
CBN	Commit comment that includes a bug report number	P
MCT	Email that closes a thread	P
MST	Email that starts a new thread	P
MFR	First reply to a thread	P
MSE	Send an email to a list	P
BRP	Report a bug	P
BDUP	Report a bug that is closed/duplicate	P
BCL	Close a bug	P
BCC	Create a comment on a bug	P

3.4 提交者网络测度插件设计与实现

提交者网络插件(Committer Network plug-in)的目标是在文件粒度上构造提交者网络. 由于Alitheia已经能够挖掘软件项目中的开发人员和文件信息，此测度插件要完成的功能是记录任意一对开发人员 A 和 B 的协同次数，即 A 和 B 添加、删除或修改的文件交集的文件数目. 图 1 给出了提交者网络插件的动态交互关系. 平台通过 Activator 注册提交者网络插件，此时 CommitterNetwork 对象将被创建. Alitheia 平台解析得到的每一个 ProjectVersion 对象都将触发 CommitterNetwork 的 getResult 来查询协同次数是否已统计至相应版本，如果没有则通过 run 方法进行统计，并添加或更新协同次数. 在 run 方法执行过程中，需要获得项目版本的提交者和变更的文件集，并查询文件集中每个文件的前一版本.

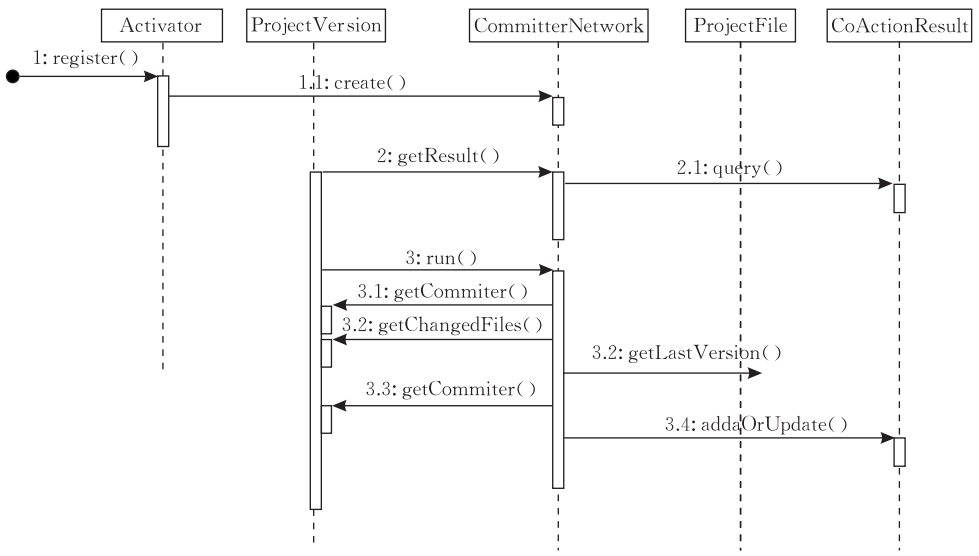


图 1 提交者网络 SDM 插件动态交互关系图

4　实验结果与分析

基于 Alitheia 平台，本实验利用贡献行为测度插件和提交者网络测度插件进行了软件资源库挖掘实验. Alitheia 平台需要在 Linux 环境运行，本实验采取主机与虚拟机之间通过 samba 协议通信进行远程开发与调试的方案.

4.1　数据收集

由于项目自身规模以及参与人员数量对实验结果会有巨大影响，因此本实验从 Gnome 开源社区中选取了具有不同项目规模（文件个数）、不同人员规模（开发人员数目）、比较成熟且得到广泛认可的 4 个软件项目作为研究对象. 项目具体信息如表 4 所示.

表 4　被选取软件项目基本信息

软件项目名	版本数	开发人员数目	最新版本文件数	SVN 库大小/MB	最新版本大小/MB
Gnome-doc-utils	1122	126	590	8.84	17.2
Gnome-VFS	5550	344	416	72.0	18.9
Gnome-utils	8123	366	570	99.8	33.8
Gnome-menus	959	144	167	5.57	2.57

4.2　开发人员贡献行为特征分析

实验中使用扩展的人员贡献行为测度插件对上述 4 个开源项目的 SVN 库进行挖掘与分析，为能够清晰地刻画人员贡献行为在不同规模项目中所具有的特征，本文中选择人员数目较多的 Gnome-VFS 与人员数目较少的 Gnome-doc-utils 两个项目进行了对比分析. 图 2 中分别显示了 Gnome-doc-utils 项目在添加代码（CADD）、删除代码（CREM）、修改代码（CCNG）、提交翻译文件（CTF）4 类行为中的人员贡献情况的柱状图（x 轴表示开发人员，y 轴表示在某类行为上的贡献度 $C_i(d_x)$）；图 3 显示了 Gnome-VFS 项目在上述四类行为中的人员行为情况的柱状图；提交二进制文件（CBF）以及无注释的提交（CEC）等行为由于出现次数极少，因此没有在本文进行展示；提交新的源代码文件（CNS）、添加新目录（CND）等行为的柱状图和（CADD）形状几乎一致，因此也没有在本文中给出.

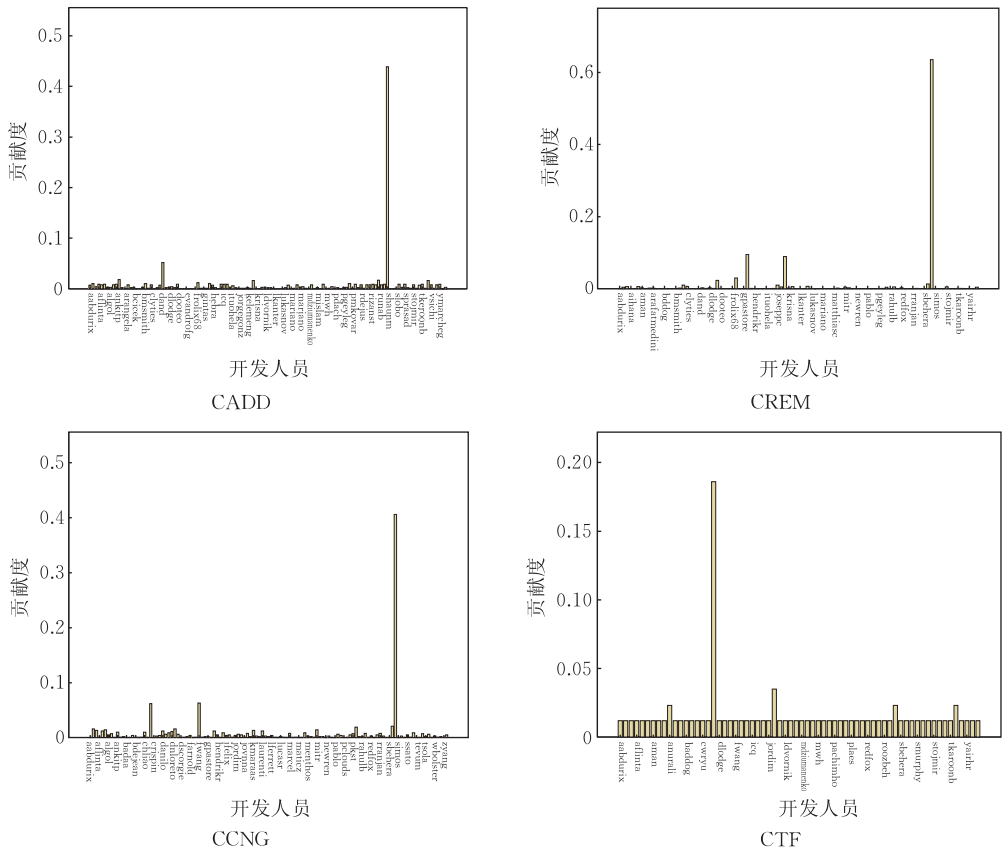


图 2　Gnome-doc-utils 4 类贡献行为柱状图

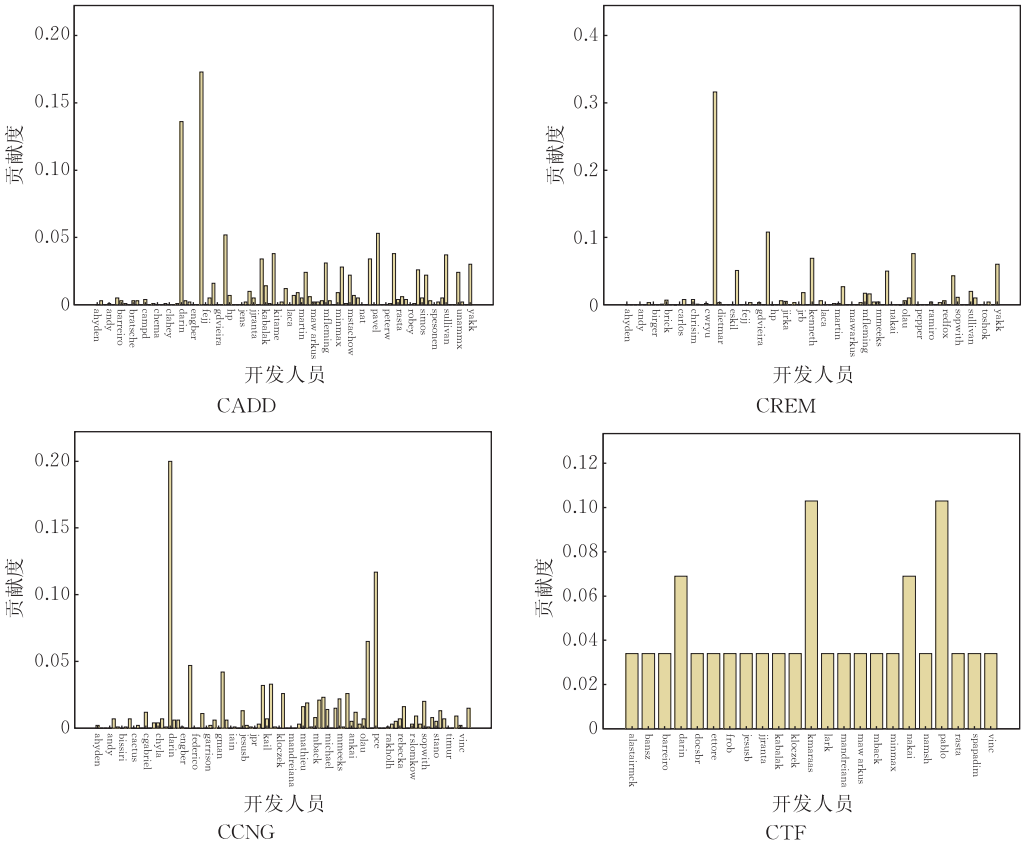


图 3 Gnome-VFS 4 类贡献行为柱状图

通过对 Gnome-doc-utils 和 Gnome-VFS 两个项目行为柱状图的分析，可以观察到如下现象，并进而得出一些关于开源开发行为的特征结论.

现象 1. 开源开发中，对于 CADD、CREM、CCNG 和 CTF 4 种行为，极少数开发人员的贡献度很高.

该现象对人员较少的 Gnome-doc-utils 来说尤为明显(如图 2)，如用户名为 shaumn 的开发人员 CADD、CREM 和 CCNG 3 种行为的贡献度都超过 40%，而其余绝大部分开发人员的贡献度不超过 3%；在 CTF 行为上，也表现出了类似现象，即其中一个开发人员的贡献度远远高于其他人员. 而对于参与开发人员数目较多的 Gnome-VFS 项目，上述行为的贡献度分布相对较分散，但从图 3 中也能够明显地区分出少数的主要开发人员，即项目核心开发人员. 该现象也从一个方面印证了 Mockus[9] 等人发现的软件开发过程中的一个重要现象，即“软件项目开发中 80%的工作由不到 20%的少数开发人员完成”.

现象 2. 对于 CADD、CREM 和 CCNG 3 种主要开发行为，在开发人员规模较小的项目(如 Gnome-doc-utils)中主要贡献者均为同一个人，而在开发人员规模较大的项目(如 Gnome-VFS)中则由不同人员承担不同开发任务.

对比 Gnome-doc-utils 和 Gnome-VFS 两个项目的 CADD、CREM 和 CCNG 3 种行为的主要贡献人员可以发现：参与人员较少的 Gnome-docs-utils 中添加、修改和删除的主要贡献者都为同一个人(用户 shaunm)，而在参与人员较多的 Gnome-VFS 中上述行为的主要贡献分别由不同人员担任. 例如：Gnome-VFS 中，在 CADD 行为上贡献度最大的开发人员 ettore(C_{ettore}(CADD)=17.3%)在 CREM 和 CCNG 两类行为上贡献度并不显著(C_{ettore}(CREM)=5.1%、C_{ettore}(CCNG)=4.7%)；反之，Gnome-VFS 中 CREM 和 CCNG 两类主要由开发人员 darin 完成，其贡献度分别为 C_{darin}(CREM)=31.6%、C_{darin}(CCNG)=20%，远远高于开发人员 ettore，而其 CADD 行为上贡献度(C_{darin}(CADD)=13.6%)又不及 ettore. 出现这种现象的原因是由于 Gnome-docs-utils 项目的开发人员规模较小，代码编写与测试任务主要由 shaunm 一个人承担；而对开发人员规模较大的 Gnome-VFS 项目来说，为对项目进行更有效的管理，需要有多个主要开发人员分别承担不同任务，如代码由 ettore 和 darin 负责代码的编写，而测试与修订工作则主要由 ettore 来完成. 由

此，从源代码开发的角度可得出如下结论：开源开发中，规模较小的项目大多具有单中心的开发模式（一般只有一个核心开发人员），而规模较大的项目一般会具有多中心的开发模式（多个核心开发人员）. 该结论与 Crowston[10] 等人利用社会网络分析的方法、基于软件缺陷修复过程得出的结论一致.

现象 3.　在 CADD、CREM 和 CCNG 等与代码开发直接相关的行为中贡献度较高的开发人员在 CTF 行为中贡献度并不突出.

在图 2 和图 3 中，通过观察 Gnome-doc-utils 和 Gnome-VFS 两个项目的 CTF 贡献行为柱状图可以发现：项目的核心开发人员并不是主要的翻译文件提交人员. 如 Gnome-docs-utils 中唯一的核心开发人员 shaunm 在 CTF 上没有任何贡献，而 Gnome-VFS 中主要开发人员 ettore 和 darin 在 CTF 行为贡献度分别只有 3.4% 和 6.9%. 出现这种现象的原因应当是由于一个软件项目从设计、开发、测试到最终成熟的过程中，需要核心开发人员完成主要的开发任务，同时也需要一些其它人员分担大量辅助性的任务，如翻译、文档、图形界面设计等工作，这些辅助性的工作往往会对项目成功与否产生巨大影响. 该现象证实了在开源开发过程中，项目的任务分工合理程度及角色配置结构对项目成长的重要影响，为本课题之前的基于开源项目角色信息的可信分析研究[5-6] 奠定了实证基础.

现象 4.　开发人员 CND、CNS 行为与 CADD 行为贡献度柱状图形状几乎一致.

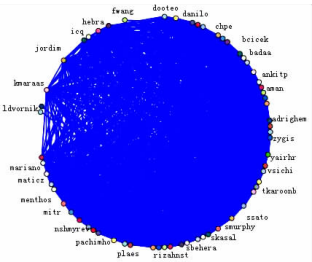
在 Gnome-doc-utils 和 Gnome-VFS 两个项目没有展示的柱状图中，提交新的源代码文件行为（CNS）、添加新目录行为（CND）与添加源代码行为（CADD）的贡献度柱状图形状几乎一致. 该现象说明了开源开发过程中，项目源代码的主要开发人员通常就是源代码文件和目录的主要创建者，也就是说，开源项目的管理员通常是由项目的核心开发人员充当.

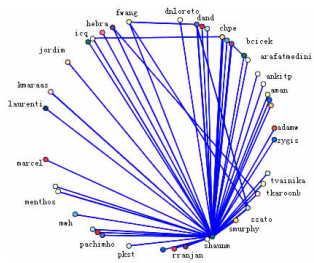
现象 5.　CBF 与 CEC 等对项目成长有负面影响的消极行为出现次数极少.

通过分析 Gnome-doc-utils 和 Gnome-VFS 两个项目可以发现，提交二进制文件（CBF）与无备注的提交（CEC）等对软件项目有负面影响的消极行为出现次数极少，如在 Gnome-docs-utils 中所有开发人员 CBF 和 CEC 行为的总和分别只有 15 和 1，而在 Gnome-VFS 中则没有发现两种行为出现. 该现象说明，在开源开发环境下，一些对项目成长有明显负面影响的行为受到显著的抑制，出现情况极少. 也就是说，开源透明的项目管理方式有益于软件项目的成长与成熟，这也是近年来开源软件获得巨大成功的主要原因之一.

4.3　提交者网络

在进一步的实验中，使用提交者网络测度插件对已选定的 4 个开源项目的 SVN 库进行分析. 本文中选取了开发人员数量相差不大，但文件个数相差较大的 Gnome-doc-utils 和 Gnome-menus 项目进行分析说明. 图 4 和图 5 描述了 Gnome-doc-utils 和


权值大等于5的边


权值大等于6的边

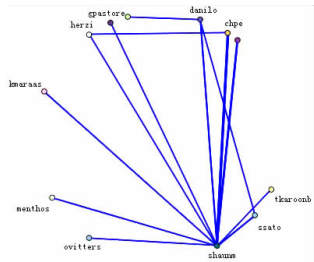
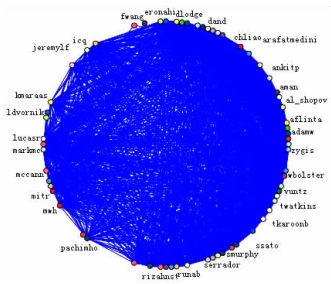
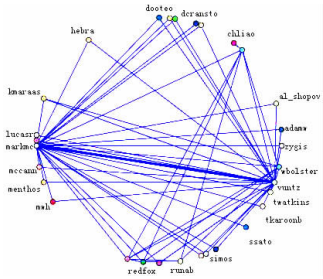

权值大等于7的边

图 4　Gnome-doc-utils 提交者网络


权值大等于5的边


权值大等于6的边

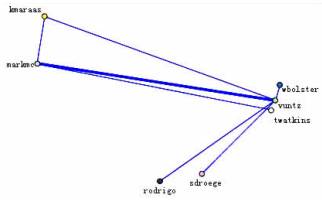

权值大等于7的边

图 5　Gnome-menus 提交者网络

Gnome-menus 项目的 3 个提交者网络结构图，图中顶点表示项目参与人员，边表示参与人员之间的协同关系，边的权重代表两个开发人员之间的协同次数，即共同参与的文件交集的个数；图 6 和图 7 分别给出了两个项目不同权重的边的分布情况（X 轴表示边的权重，Y 轴表示边的个数）. 通过观察分析，可得到如下现象与结论.

现象 6. 权重为 2 的边出现次数最多，权重大于 5 的协同行为几乎很少出现.

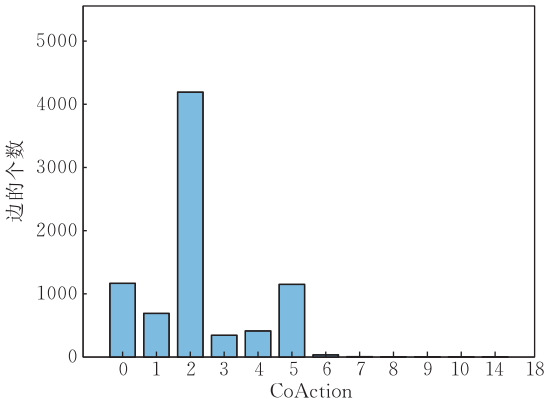


图 6 Gnome-doc-utils 提交者网络不同权值边的个数

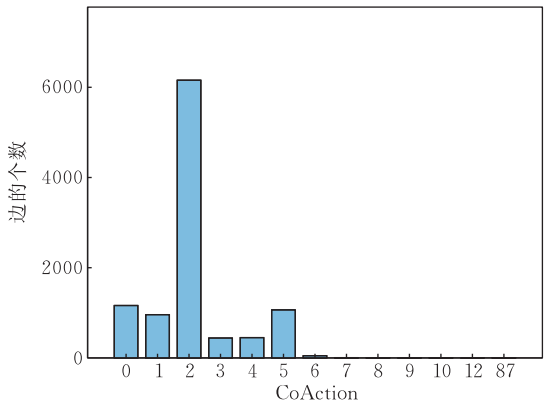


图 7 Gnome-menus 提交者网络不同权值边的个数

通过图 6 和图 7 中可以发现，Gnome-doc-utils 项目的提交者网络中边的最大权值为 18，而在 Gnome-menus 中最大权值为 87. 虽然两个项目文件数目差距很大（Gnome-doc-utils 为 590 个，Gnome-menus 为 167 个），但可以清晰地看出图 6 和图 7 的形状几乎一致. 从两个项目的提交者网络中都可以看出，满足 $w_i \geq 6$ 的边的个数明显少于 $w_i \geq 5$ 边，这说明权重为 5 的协同行为在两个项目中普遍存在，而协同次数大于 5 的协同行为则很少出现. 另外，权重为 2 的协同行为出现次数最多，而大于 5 的协同行为则几乎很少出现. 这说明在项目开发过程中，开发人员共同负责 2 个文件的协同行为是最常见的情况，而共同负责 6 个文件以上的协同行为极少出现.

现象 7. 项目整个开发过程中，主要开发人员频繁参与协同行为.

根据人员贡献行为测度插件的计算结果，可以得到每个开发人员在添加、删除、修改代码等行为（CADD、CREM、CCNG）上的贡献度，表 5 中给出了 Gnome-doc-utils 和 Gnome-menus 项目中主要开发人员在这 3 类行为上的贡献度. 由于其他人员在这些行为上的贡献度普遍不到 3%，因此可以得出结论，shaumn 和 markmc 两人分别是两个项目的核心开发人员；由于 vuntz 在删除代码（CREM）行为上有较高的贡献度，因此也可以认为他是项目的主要开发人员之一. 在两个项目的提交者网络中（图 4 和图 5）可以清晰地看出，这 3 个开发人员结点上有大量的边存在，这表明他们参与了项目开发过程中大量的协同行为. 其中，Shaumn 与其他人员的总协同次数为 1121（与该结点相连的边权重的总和），而 vuntz 和 markmc 分别为 723 和 744，均远远高于参与项目的其他成员的协同次数. 该现象表明：在开源开发模式下，核心开发人员会频繁地参与项目协同行为，即核心开发人员会负责协调与控制整个项目的进展.

表 5 项目主要开发人员贡献度

开发人员	添加代码/%	删除代码/%	修改代码/%	所属项目
shaumn	4.39	63.5	40.6	Gnome-doc-utils
markmc	4.12	54.1	27.8	Gnome-menus
vuntz	3.8	16.5	3.6	Gnome-menus

现象 8. 项目主要开发人员之间存在频繁的协同行为，外围开发人员之间却几乎很少参与项目协同工作.

在 Gnome-menus 的提交者网络中，主要开发人员 markmc 与 vuntz 之间的协同次数高达 87，而外围开发人员之间的协同次数几乎都不大于 7. 该现象符合 Huang 等人发现的结果，即项目主要开发人员之间存在频繁的协同行为，而外围开发者之间却几乎很少参与合作.

5 结论和下一步工作

如何刻画软件项目开发人员的行为特征是软件工程领域所关注的问题之一，本文通过对参与项目的开发人员多种贡献行为的度量和协同行为的分

析，揭示了开源环境下开发人员的部分行为特征，发现了一些具有重要理论和实践价值的现象. 在分析过程中，基于 Alitheia 软件资源库挖掘平台扩展并实现了与研究问题相关的测度插件，即开发人员贡献行为测度插件及提交者网络构造插件. 在本文的工作中，目前的分析都是基于版本控制系统 SVN 资源库进行的，尚没有涉及到漏洞跟踪系统以及邮件列表库中的人员行为信息，在今后工作中，我们将设计并进行更大规模的分析与实验，综合考虑多种类型的软件资源库中的人员贡献行为，为进一步揭示与软件开发模式、软件可信性等相关的证据信息做出更深入研究. 另外，虽然 Alitheia 平台具有合理的架构，提供了良好的可扩展性，但平台本身尚未正式发布，不够成熟. 基于目前已建立的与 Alitheia 开发团队之间的联系，本课题计划以开源方式对该项目进行进一步的参与和完善，针对其目前的性能和功能缺陷做出改进.

参　考　文　献

[1] Sackman H, Erikson W J, Grant E E. Exploratory experimental studies comparing online and offline programming performance. Communications of the ACM, 1968, 11(1): 3-11

[2] Gousios G, Kalliamvakou E, Spinellis D. Measuring developer contribution from software repository data//Proceedings of the 5th International Working Conference on Mining Software Repositories. Leinzig, 2008: 129-132

[3] Lopez-Fernandez L, Robles G, González Barahona J M. Applying social network analysis to the information in CVS repositories//Proceedings of the 1st International Workshop on Mining Software Repositories. Edinburgh, 2004: 101-105

[4] Huang Shikun, Liu Kangmin. Mining version histories to verify the learning process of legitimate peripheral participants//Proceedings of the 2nd International Workshop on Mining Software Repositories. Saint Louis, 2005: 84-87

[5] Yuan Lin, Wang Huai-Min, Yin Gang, Shi Dian-Xi, Mi Hai-Bo. Mining roles of open source software//Proceedings of the 2nd International Conference on Software Engineering and Data Mining. Chengdu, 2010: 548-554

[6] Yuan Lin, Wang Huai-Min, Yin Gang, Shi Dian-Xi, Mi Hai-Bo. Trustworthy evaluation technology of software based on roles. Journal of Beijing University of Technology, 2010, 36(5): 611-615(in Chinese)
(袁霖，王怀民，尹刚，史殿习，米海波. 基于角色的软件可信评估技术研究. 北京工业大学学报，2010，36(5)：611-615)

[7] Gousios G, Spinellis D. A platform for software engineering research//Proceedings of the 6th International Workshop on Mining Software Repositories. Vancouver, 2009: 31-40

[8] Gousios Georgios. Tools and methods for large scale empirical software engineering research [Ph. D. dissertation]. Athens University of Economics and Business, Athens, 2009

[9] Mockus A, Fielding T, Herbsleb D. Two case studies of open source software development: Apache and Mozilla. ACM Transactions on Software Engineering and Methodology, 2002, 11(3): 309-346

[10] Crowston Kevin, Howison James. The social structure of open source software development. Journal of Internet, 2005, 10(2): 132-141

YUAN Lin, born in 1981, Ph. D. candidate, lecturer. His current research interests include data mining and trustworthy evaluation of software.

WANG Huai-Min, born in 1962, Ph. D., professor, Ph. D. supervisor. His current research interests include distributed objected oriented technologies and trustworthy software.

YIN Gang, born in 1975, Ph. D., lecturer. His current research interests include distributed objected-oriented technologies and network security.

SHI Dian-Xi, born in 1966, Ph. D., associate professor. His research interests include distributed computing and autonomic computing.

LI Xiang, born in 1988, M. S. candidate. His current research interests focus on data mining technology.

Background

This research is supported by the National High Technology Research and Development Program (863 Program) of China under grant (2007AA010301), National Natural Science Foundation of China under grant (No. 60903043), and the "Core Electronic Devices, High-End General Chip and Fundamental Software" Major Project (2009ZX01043-001).

Since the software appeared, how to assess the quality of software objectively and efficiently and prevent huge losses

caused by software problems has been one of the core issues in software engineering. Therefore the technology of produce and evaluation of high trustworthy software has become an important research field in recent years. Trusted software usually refers to the software whose behavior and results are always consistent with peoples' expectation and would be able to provide continuous service even if it was interfered. With the development of the Internet and appearance of trusted software conception, the technology of software trustworthy evaluation became an urgent problem in the field of software trustworthy research. But how to determine one software project is trustworthy and how to measure the level of software trustworthy are the core issues in software trustworthy evaluation. Software trustworthiness evidence is the evidence which could be validated and can be used to affirm the facts of software trustworthy. It is the important part of software trustworthy evaluation, and can be the direct basis to assure the level of software trustworthy. During the whole life cycle of software, a very large amount of data with various types is produced. In recent years, more and more researchers in the world realized the great value of these data. In the process of their study about the software trustworthy evaluation, the authors come to realize these data may provide some new ideas for software trustworthy assessment, yet up to now they still have not been made full use of in the field of software trustworthy assessment. The work in this paper prepares the ground for quantification and utilization of the software trustworthiness evidence from the data in software repository and is part of the research on evaluation of software projects. The authors expect to select some kind of data from software repository which can be used to improve or even replace some metrics in the existing evaluation models. The work will help to evaluate software projects efficiently and automatically. In this paper, the authors mostly discussed the behavioral characteristic of developers.