(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 104598375 B
(45)授权公告日 2017.09.12

(21)申请号 201410710095.6

(22)申请日 2014.11.28

(65)同一申请的已公布的文献号
申请公布号 CN 104598375 A

(43)申请公布日 2015.05.06

(73)专利权人 江苏苏测软件检测技术有限公司
地址 210000 江苏省南京市雨花台区宁双路28号汇智大厦7楼
专利权人 南京大学

(72)发明人 周骏贵 陈振宇 张伟强 张驰 濮力 程秀才 谢佩章 王婧宇

(74)专利代理机构 南京天翼专利代理有限责任公司 32112
代理人 奚铭

(51)Int.Cl.
G06F 11/36(2006.01)
G06F 17/30(2006.01)

(56)对比文件
CN 101866316 A,2010.10.20,
US 2010/0058300 A1,2010.03.04,
Bettenburg 等.Studying the Impact of Social Structures.《Proceedings of the 2010IEEE 18th International Conference on Program Comprehension,Braga,Portugal》.2010,第124页第I节-第132页第VII节.

审查员 林坚

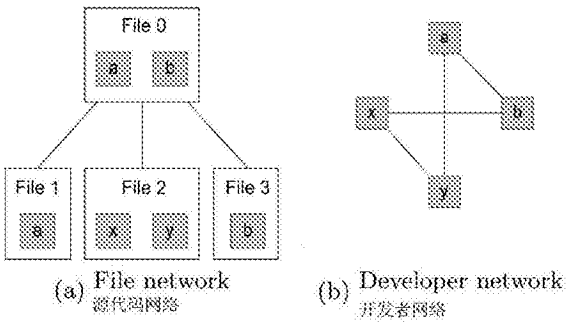
权利要求书1页 说明书4页 附图1页

(54)发明名称

一种用于软件开发的缺陷预测方法

(57)摘要

一种用于软件开发的缺陷预测方法，构建源代码依赖网络和开发者社交网络，分别记录源代码文件间的依赖关系，以及开发者间的交流协作关系。比较源代码依赖网络和开发者社交网络，将两者的不一致作为缺乏交流的指示。本发明可以检测出每个源代码文件关联的开发者交流链接缺乏的数量，作为衡量文件网络不一致性的指标，还可以给出文件缺乏的开发者链接，并同时给出每一条链接缺乏的原因，即哪两个源文件之间存在依赖，而其开发者间没有交流协作。开发者可以根据方法的指导，组织弥补相应的交流，有针对性地发现和修复缺陷。



CN 104598375 B

1.一种用于软件开发的缺陷预测方法,其特征是包括以下步骤:

1)构建源代码依赖网络和开发者社交网络,源代码依赖网络根据源代码的依赖关系建立,用于记录源代码文件之间的依赖关系,开发者社交网络根据软件开发者的是否参与过同一个开发任务来建立,用于记录开发者间的交流协作关系;

其中,源代码依赖网络包含多种代码关系,开发者社交网络包括多种开发者关系:

源代码依赖网络包括:

代码语法依赖:静态分析各个源代码文件间的数据依赖和控制依赖关系,建立文件间的语法依赖网络;

代码逻辑依赖:分析过程数据,找出同时被提交,即一起被修改的文件,开源项目的版本控制工具中记录了每一次修改所提交的文件,通过分析每一次修改所提交的文件得到文件间的逻辑依赖网络;

代码任务依赖:找出为了完成同一个任务而同时修改的文件,建立这些文件间的任务依赖关系;

开发者社交网络包括:

开发者任务依赖:两个开发者同时参与了一个任务,则存在任务依赖关系;

开发者提交重叠:两个开发者提交过同一个文件,则建立该关系;

2)基于"源代码存在依赖关系,则相应开发者间需要有社交关系"的原则,寻找源代码依赖网络和开发者社交网络间的不一致,所述不一致即缺乏的开发者关系;

3)将每个源代码文件所关联的缺乏的开发者关系作为源代码文件不一致性的度量值,用于源代码文件的缺陷预测,缺乏的开发者关系越多,源代码文件存在潜在缺陷以及未来需要修复的可能性越高。

2.根据权利要求1所述的一种用于软件开发的缺陷预测方法,其特征是还包括步骤4):

4)在步骤3)进行缺陷预测的同时,根据缺乏的开发者关系提取对应的源代码,所提取的源代码即为可能存在缺陷的部分。

3.根据权利要求1或2所述的一种用于软件开发的缺陷预测方法,其特征是所述步骤2)具体包括如下步骤:

21)根据源代码网络中的依赖关系,以及源代码文件和开发者的提交关系,认定存在依赖关系的源代码文件的开发者间应具有必要的社交关系;

22)检查步骤21)判定的开发者间的必要社交关系是否在开发者社交网络中存在;

23)步骤22)检查中不存在的社交关系被认定为缺乏的开发者社交关系链接,即缺乏的开发者关系。

一种用于软件开发的缺陷预测方法

技术领域

[0001] 本发明属于计算机技术领域，涉及IT技术中的软件工程领域，用于改善软件开发团队内部交流沟通网络的环节缺失，为一种用于软件开发的缺陷预测方法。

背景技术

[0002] 软件开发是一项极为依赖开发者人工工作的智力活动，其中开发者之间的交流和协作对软件质量有着极大影响。很多严重的软件质量问题根源上都是由于缺乏充分的交流和沟通导致的，从而造成软件成本的昂贵。

[0003] 缺陷预测是近些年学术界众多研究人员关注的领域，目的是提前预测项目代码中哪些模块或文件最有可能出现缺陷，从而提前分配资源对最有威胁的部分采用静态审查及动态测试等质量保障手段发现潜在的缺陷。目前主要的技术是采用各种机器学习算法，通过采用源代码和修改日志等数据中计算得到的各种度量值组合成输入参数，建立回归模型进行预测，计算出各个模块或文件的缺陷可能性指标。

[0004] 然而这样的技术在工业中却难以得到有效应用。原因是研究中隐含设定了“完美开发者”的不合理假设，认为把未来会发生缺陷的文件呈现给开发者，开发者就能立即发现其中的缺陷。事实上，当开发者看到缺陷预测给出的文件列表时，常常会感到困惑，不知道从何入手，因此会直接忽视预测结果。因为当前基于机器学习算法和统计分析模型的缺陷预测技术无法给出有效的指导，不能告诉开发者代码具体有怎样的问题和解决建议。

[0005] 在现有的缺陷预测领域，目前的技术只考虑表征，而不考虑引发缺陷的根源，因此预测得到的结果难以给开发者有效帮助。

发明内容

[0006] 本发明要解决的问题是：在软件开发过程中需要能够自动发现必要却缺乏的开发者间的交流，即发现软件开发流程中的缺失环节，同时给出受到交流缺乏的影响而可能存在缺陷的相应源代码文件。

[0007] 本发明的技术方案为：一种用于软件开发的缺陷预测方法，包括以下步骤：

[0008] 1)构建源代码依赖网络和开发者社交网络，源代码依赖网络根据源代码的依赖关系建立，用于记录源代码文件之间的依赖关系，开发者社交网络根据软件开发者的是否参与过同一个开发任务来建立，用于记录开发者间的交流协作关系；

[0009] 2)基于“源代码存在依赖关系，则相应开发者间需要有社交关系”的原则，寻找源代码依赖网络和开发者社交网络间的不一致，所述不一致即缺乏的开发者关系；

[0010] 3)将每个源代码文件所关联的缺乏的开发者关系作为源代码文件不一致性的度量值，用于源代码文件的缺陷预测，缺乏的开发者关系越多，源代码文件存在潜在缺陷以及未来需要修复的可能性越高。

[0011] 进一步的，还包括步骤4)：

[0012] 4)在步骤3)进行缺陷预测的同时，根据缺乏的开发者关系提取对应的源代码，所

提取的源代码即为可能存在缺陷的部分。

[0013] 步骤1)中，源代码依赖网络包含多种代码关系，开发者社交网络包括多种开发者关系：

[0014] 源代码依赖网络包括：

[0015] 代码语法依赖：静态分析各个源代码文件间的数据依赖和控制依赖关系，建立文件间的语法依赖网络；

[0016] 代码逻辑依赖：分析过程数据，找出同时被提交，即一起被修改的文件，开源项目的版本控制工具中记录了每一次修改所提交的文件，通过分析每一次修改所提交的文件得到文件间的逻辑依赖网络；

[0017] 代码任务依赖：找出为了完成同一个任务而同时修改的文件，建立这些文件间的任务依赖关系；

[0018] 开发者社交网络包括：

[0019] 开发者任务依赖：两个开发者同时参与了一个任务，则存在任务依赖关系；

[0020] 开发者提交重叠：两个开发者提交过同一个文件，则建立该关系。

[0021] 上述步骤2)具体包括如下步骤：

[0022] 21)根据源代码网络中的依赖关系，以及源代码文件和开发者的提交关系，认定存在依赖关系的源代码文件的开发者间应具有必要的社交关系；

[0023] 22)检查步骤21)判定的开发者间的必要社交关系是否在开发者社交网络中存在；

[0024] 23)步骤22)检查中不存在的社交关系被认定为缺乏的开发者社交关系链接，即缺乏的开发者关系。

[0025] 现有技术中，在研究和工业中都忽视了开发者间缺乏交流会引发缺陷这一点问题，而且因为交流缺乏这一问题难以采用自动化的方式被高效地发现，因此也没有人基于这一点来进行缺陷预测。

[0026] 根据本发明的技术方案，预测了软件的缺陷风险，给出源代码依赖网络和开发者社交网络不一致性的同时，还可以给出文件缺乏的开发者链接，并同时可以由源代码依赖网络和开发者社交网络的内容明确每一条链接缺乏的原因，即哪两个源文件之间存在依赖，而其开发者间没有交流协作。开发者可以根据方法的指导，组织弥补相应的交流，有针对性地发现和修复缺陷。

附图说明

[0027] 图1为本发明一个示例，显示了本发明的实施方式，左半边给出了一个源代码网络的一部分，包含4个文件及其依赖关系，文件中阴影区域表示该文件的开发者；右半边给出了这些开发者在开发者网络间的社交关系。

[0028] 图2为根据图1给出的关系计算File 0的网络不一致性度量值的过程。

具体实施方式

[0029] 本发明是一种为改善软件开发团队内部交流沟通提供自动化推荐的方法，从而辅助软件团队更好地完成软件开发，大大减少因为成员间不充分的交流造成的软件质量问题。本发明可以实现为自动化工具，部署在软件项目，尤其是开源项目或者地理分散的团队

的开发平台中，帮助技术负责人了解软件团队的交流情况，并及时修复因交流不充足导致的软件缺陷。

[0030] 本发明的核心技术是比较源代码依赖网络和开发者社交网络，将两者的不一致作为缺乏交流的指示。

[0031] 首先构建源代码依赖网络和开发者社交网络。目前有各种方式构建这两种网络，本发明中将这些方法组合起来，最终得到一个合成的源代码依赖网络和一个合成的开发者社交网络，分别记录了源代码文件间的依赖关系，以及开发者间的交流协作关系。

[0032] 接下来寻找代码网络和开发者网络间的不一致。我们认为，如果两个源文件间存在依赖，那么其各自的开发者之间在很大程度上应该有交流和协作，否则就是出现了不一致，交流链缺乏。按照这一设定，本发明可以找到很可能缺乏而又必要的开发者链接。

[0033] 最终本发明可以检测出每个文件关联的开发者交流链接缺乏的数量，作为衡量文件网络不一致性的指标。

[0034] 下面给出以真实的开源项目作为实验主体验证本发明效果所进行的实验步骤，供实践者在进行具体实施时加以参考，也可根据具体项目环境和特点灵活调整。

[0035] 1)构建网络

[0036] 两大类网络的构建方式目前有很多，实验中我们在构建它们时分别采用了三种方法。

[0037] 源代码依赖网络：

[0038] ●代码语法依赖：静态分析各个源代码文件间的数据依赖和控制依赖关系。实验中使用了Understand工具完成这一工作，建立文件间的语法依赖网络。

[0039] ●代码逻辑依赖：分析过程数据，找出同时被提交，即一起被修改的文件。开源项目的版本控制工具如SVN、Github等系统中记录了每一次修改所提交的文件，可以分析这些数据得到文件间的逻辑依赖网络。

[0040] ●代码任务依赖：找出为了完成同一个任务(如修复一个缺陷或者实现一项需求)而同时修改的文件，建立关系。实验中的开源项目版本控制工具提交的消息中会附有某一次提交是为了完成哪个任务而进行的，与缺陷跟踪系统结合起来可以分析得到这些提交所包含的文件间存在任务依赖关系。

[0041] 开发者社交网络：

[0042] ●开发者任务依赖：两个开发者同时参与了一个任务，则存在任务依赖关系。类似于代码任务依赖关系的提取，实验中找出不同的提交用于完成同一项任务，但其开发者不同，则建立任务依赖关系。

[0043] ●开发者提交重叠：两个开发者提交过同一个文件，则建立该关系。从版本控制工具记录的过程日志上可以找到。

[0044] ●开发者评论交流：两个开发者对同一个任务发表过评论，则建立该关系。从缺陷跟踪系统中可以获取该信息。

[0045] 以上方式是研究中已经经过反复检验过的方法，本发明在实施例中将三种代码网络构建方法构建得到的网络合成一个，将三种开发者社交网络也合成一个，来展开后面的分析。或者说，无论哪一种代码依赖关系，本发明首次提出，如果与代码相对应的开发者没有任何一种社交关系，本发明方法就认为他们缺乏交流。()本发明实施中也可尝试采用其

他的关系网络构建方法。

[0046] 在网络构建中常常会遇到一个困难，版本控制工具中提交代码的开发者，和缺陷跟踪系统(报告、跟踪、管理缺陷的系统，在软件开发团队中广泛使用)中修复和评论Bug的开发者难以映射，这可能需要一些数据预处理的工作(使用例如启发式规则等方法将版本控制工具和缺陷跟踪系统中的开发者映射起来)，或者需要依靠成熟的项目管理平台以及使用方式(将版本控制系统和缺陷跟踪系统集成起来，无缝对接)。

[0047] 本发明实施例中构建网络采用的数据是以版本为单位的，即每一个版本的开发周期内产生的数据用于构建一套网络。实施时也可按照具体情况进行调整。

[0048] 2)计算度量值

[0049] 建立起源代码依赖网络和开发者社交网络后，接下来比较二者，寻找不一致。对于每个文件，计算其网络不一致性，即缺乏的开发者交流数量的方法如下：

[0050] 1.找出所有文件f的开发者，根据版本控制系统中的记录可知道开发者提交过哪些文件，放入集合D(f)；

[0051] 2.在源代码依赖网络中找出与文件f相关联的文件，放入集合F(f)；

[0052] 3.找出F(f)中每个源代码文件的所有开发者，放入集合D(F(f))；

[0053] 4.把D(f)和D(F(f))两个集合中的开发者两两相连，每一对关联都作为必要的开发者链接放入NDL集合；

[0054] 5.对于NDL集合中的每一条链接，在开发者社交网络中查找其是否存在，存在则放入存在的开发者链接集合EDL；

[0055] 6.不存在则放入缺失的开发者链接集合MDL，MDL为NDL与EDL的差集；

[0056] 7.MDL的模即缺失的开发者链接数，也是文件f的网络不一致性度量值。

[0057] 图1和附图2举例说明了计算度量值的方法。

[0058] 得到了每个文件的度量值后，团队可以选出网络不一致性度量大于0的文件作为可疑文件进行审查，或者先从不一致性最高的文件开始审查。对于每个文件，方法不仅可以提供一个整数值，还可以给出每一条缺乏的开发者链接，是哪两个开发者之间出现了交流缺乏情况；更重要的是给出其原因，即因为哪两个文件存在依赖，而导致这两个开发者被认定为需要进行交流协作的开发者。有了这些信息，团队负责人可以召集相关开发者进行讨论，看是否确实存在问题，并加以弥补和修复。

[0059] 以上实施方案可以集成到项目开发团队使用的管理平台中，如开源项目常用的Github、Trac等，按时为团队负责人自动化地展示分析结果。

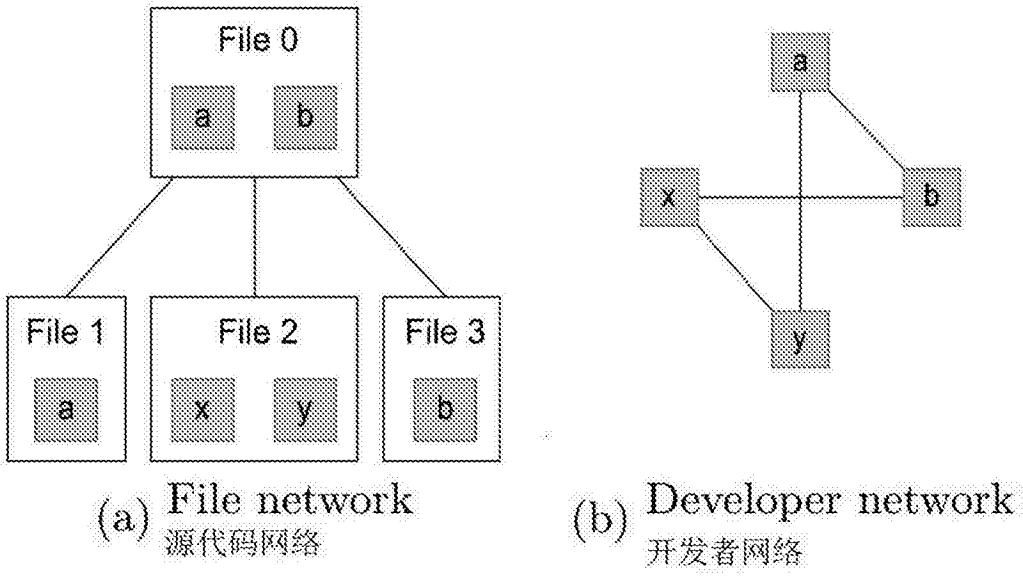


图1

Step 1 (步骤1) Developers of $File_0$:

$$D(file_0) = \{a, b\}.$$

Step 2 (步骤2) Files connected to $File_0$:

$$F(file_0) = \{file_1, file_2, file_3\}.$$

Step 3 (步骤3) Developers of connected files:

$$D(\{file_1, file_2, file_3\}) = \{a, b, x, y\}.$$

Step 4 (步骤4) Necessary Developer Links (all the possible links between $\{a, b\}$ and $\{a, b, x, y\}$):

$$NDL(file_0) = \{(a, b), (a, x), (a, y), (b, x), (b, y)\}.$$

Step 5 (步骤5) Existing Developer Links in the developer network:

$$EDL(file_0) = \{(a, b), (a, y), (b, x)\}.$$

Step 6 (步骤6) Missing Developer Links:

$$MDL(file_0) = \{(a, x), (b, y)\}.$$

Step 7 (步骤7) Inconsistency:

$$\mathcal{I}(file_0) = \left|\{(a, x), (b, y)\}\right| = 2.$$

图2