Github 课题前沿论文最新进展

2018.11.06 方建勇

提示：采用手机 safari 微软翻译技术

1. **建议: 1810.13062[pdf,其他]** si

   羽毛羊群的鸟在一起？github 和堆栈溢出中开发商的植绒和迁移行为研究

   作者:michael musun, akash ghosh, rajesh sharma, sandeep kaur kuttal

   摘要: 个人与参与社区活动之间的互动取决于个人如何与同龄人认同。我们希望在开发人员在社交协作环境 (特别是代码托管网站和问题回答网站) 上学习和提供帮助时，为他们调查此类行为。在本研究中，我们将调查以下有关倡导者、开发人员的问题，他们可以被确定为全面的社区贡献者和活跃的学习者。倡导者是否聚集在一个社区里？成群结队的倡导者如何在社区内迁移？这些倡导者群体是否超越了一个社区？为了了解这种行为，我们在一个代码托管站点-github 和一个问题回答站点-堆栈溢出收集了 12 578 名常见的提倡者。这些倡导者参与了 github 上的 1 549 个项目，并积极询问了 114 569 个问题，并就堆栈溢出问题回答了 40858 项答复和 1, 001 125 条评论。我们利用社交网络进行了深入的实证分析，以发现在 github、堆栈溢出和两个社区之间的倡导者群及其迁移模式。我们发现, 7.5% 的倡导者在 github 上制造羊群, 8.7% 在堆栈溢出。此外，这些倡导者群平均在 github 上迁移 5 次，在堆栈溢出上迁移 2 次。特别是，两个羊群中的倡导者比较大的羊群迁移的频率更高。但是，这种迁移行为仅在单个社区中常见。我们的研究结果表明，倡导者的蜂拥而至和迁移行为与其他社会环境中的羊群和迁移行为有很大不同。这表明有必要调查影响倡导者的蜂拥而至和迁移行为的因素，以及如何在协作软件工具中增强和整合对此类行为的支持。少

   2018 年 10 月 30 日提交;最初宣布 2018 年 10 月。

2. **第: 1810.4825[pdf]** cse

   多伊 10.1109/ACCESS.2018.2872669

   基于 github 存储库的器官移植初探

   作者:王尚文,毛晓光,俞觉敏

   文摘 器官移植是利用与某些特定功能直接相关的代码来完成自己的程序，它比传统的组件重用为开发人员提供了更多的便利。然而，最近的技术面临着缺乏移植器官的挑战。因此，我们对从 github 存储库中提取器官进行了实证研究，以探索基于大规模数据集的移植。我们分析了 12 个具有代表性的 github 项目的统计数据，得出的结论是: 1) 在提交中存在大量的实际器官，并在评论中添加一个关键词;2) 本资料库中的机关主要有四种内容;3) 大约 70% 的器官易于移植。实施不同类型器官的移植策略，我们用三种不同的编程语言 (java、python 和 c) 手动提取 30 个器官，并利用四种测试工具 (两个用于 java，一个用于 python，和一个为 c)。最后，我们将三个 java 器官移植到一个特定的平台上进行性能检查，以验证它们是否能在新系统中正常工作。我们的策略提取的 30 个器官在单元测试中都有良好的性能，最高合格率达到 97%，最低的器官仍然通过 80%，三个 java 器官在新系统中工作良好，为主机提供了三个新的功能。结果表明，基于开源存储库的器官移植是可行的，为代码重用带来了新的思路。少

2018 年 10 月 10 日提交;最初宣布 2018 年 10 月。

评论:14 页

3. 第 xiv:1809.04041[pdf,其他] cse

在 github 中识别未维护的项目

作者:jailton coelho, marco tulio valente, lu 西亚 a l.silva, emad shihab

摘要 背景: 开源软件在现代软件开发中的重要性日益增加。然而,人们对这类项目的可持续性也越来越感到关切,这些项目通常由少数开发商管理,经常担任志愿者。目标:在本文中,我们提出了一种方法来识别没有积极维护的 github 项目。我们的目标是提醒用户使用这些项目的风险,并可能激励其他开发人员承担项目的维护。方法: 我们根据一组有关项目活动的功能 (提交、叉、问题等),训练机器学习模型,以识别不维护或维护稀少的项目。我们与 129 个 github 项目的主要开发人员以最佳性能对模型进行了经验验证。结果: 基于实际开源开发者的反馈,提出的机器学习方法的精度为 80%;和 96% 的召回。我们还表明,我们的方法可以用来评估项目得不到维护的风险。结论:本文提出的模型可供开源用户和开发人员用于识别不再积极维护的 github 项目。少

2018 年 9 月 11 日提交;最初宣布 2018 年 9 月。

评论:2018 年在第十二届经验软件工程与测量国际研讨会 (esem) 上接受, 10 页

4. 第 xiv:809.00 580[pdf,其他] cse

多伊 10.1109/FIE.2017.8190589

ci 教授: 采用持续集成服务和 github 工作流来教授测试驱动开发

作者:christoph matthies, arian treffer, Matthies uflacker

摘要: 使用大规模开放在线课程 (mooc) 的教学编程由于其可扩展性和知识分发的效率而越来越受欢迎。但是,参加这些课程通常意味着完全致力于浏览器中提供的编程环境。虽然这样可以实现一致和可控的设置,但学习者不会获得实际开发工具的经验,如本地代码编辑器、测试框架、问题跟踪器或持续集成 (ci) 服务,这对后续开发工具至关重要现实世界中的项目。此外,即将开发的功能测试通常在 mooc 中可用,只需执行,从而减少了对开发适当测试的参与。为了在保持高度自动化和可扩展性的同时解决这些问题,我们开发了 ci 教授,这是一种进行在线练习的新方法。ci 教授利用开发人员日常使用的现有自动化基础架构 (即 ci 服务和 github 工作流) 来教授测试驱动开发 (tdd) 实践。参与者在 github 中使用自己的存储库,并在推送代码时收到来自 ci 服务器的反馈和新挑战。我们已成功地将此方法应用于一个试点项目中, 30 名本科生学习 ruby on rails web 开发框架。我们的评估表明,这项工作有效地提高了学生为代码写测试的动力。我们还介绍了参与者调查的结果、学生的经历和教师的观察。少

2018 年 9 月 3 日提交;最初宣布 2018 年 9 月。

期刊参考: 2017 年 ieee 教育前沿会议 (fie),印第安纳波利斯, in, 2017, 第 1-8 页

5. 第 xiv:1808. 04919[pdf, ps,其他] cse

giststgithb 上 python 代码段的可执行性的评估

作者:eric horton, chris parnin

摘要: 软件开发人员在线创建和共享代码,以演示编程语言概念和编程任务。代码段可以是解释和演示编程概念的有用方法,但可能并不总是可以直接执行的。代码段可能包含分析错误,或者在环境包含未满足的依赖项时无法执行。本文对通过 github gist 系统共享的 python 代码段的可执行状态以及熟悉软件配置的开发人员正确配置和运行

它们的能力进行了实证分析。我们发现, 75.6 的专家需要非平凡的配置来克服缺少的依赖项、配置文件、对特定操作系统的依赖或其他一些环境配置。我们的研究还表明, 当解决配置错误不到一半的时间时, 开发人员对资源名称的自然假设是正确的。我们还提出了 gistable, 一个数据库和可扩展的框架, 建立在 github 的 gist 系统上, 它提供了可执行的代码片段, 以便能够在软件工程中进行可重复的研究。Gistable 包含 10, 259 个代码段, 大约 5, 000 个代码段, 其中包含一个 dokerfile 文件, 用于配置和执行它们, 而不会出现导入错误。Gistable 于 https://github.com/gistable/gistable 公开提供。少

2018 年 8 月 14 日提交;最初宣布 2018 年 8 月。

6. **第 1807. 04130[pdf,其他] cse**

多伊 10.114/2970272970283

更正: vendasta 技术在 github 的代码审阅者推荐

作者:mohammad masudur rahman, chanchal k.roy , jesse redl , jason a. collins

摘要: 对等代码评审在软件开发的早期阶段定位常见的编码标准冲突和简单的逻辑错误, 从而降低总体成本。遗憾的是, 在 github, 为请求识别适当的代码审阅者具有挑战性, 因为通常无法随时获得用于审阅者标识的可靠信息。在本文中, 我们提出了一个代码审阅者推荐工具--corect--它不仅考虑了开发人员的相关跨项目工作经验 (例如外部库经验), 还考虑了他们在某些专业技术方面的经验 (例如,谷歌应用程序引擎) 与一个拉请求, 以确定她作为一个潜在的代码审查员的专业知识。我们使用客户端-服务器体系结构设计我们的工具, 然后将解决方案打包为 google chrome 插件。一旦开发人员在 github 启动新的拉请求, 我们的工具将自动分析该请求, 挖掘两个相关历史记录, 然后返回浏览器上下文中请求的相应代码审阅者的排名列表。演示: https://www.youtube.com/watch? v=rXU1wTD6QQ0 少

2018 年 7 月 9 日提交;最初宣布 2018 年 7 月。

评论:第三十一届 ieeeacm 自动化软件工程国际会议 (ase 2016), 第 792-797 页, 新加坡, 2016 年 9 月. arxiv 行政说明: 文本与 arxiv:1807. 2965 有实质性重叠

日记本参考:proc. ase 2016, 第 792 页----792 页

7. **第 xiv:1807. 02965[pdf,其他] cse**

多伊 10.11145/288916. 2889244

正确: 基于跨项目和技术经验的 github 代码审阅者推荐

作者:mohammad masudur rahman, chanchal k. roy,jason a. collins

摘要: 对等代码评审在软件开发的早期阶段定位常见的违反编码规则和简单的逻辑错误, 从而降低总体成本。但是, 在 github 中, 为请求标识适当的代码审阅者是一项重要的任务, 因为评审器标识的可靠信息通常不容易获得。在本文中, 我们提出了一种代码审阅者推荐技术, 该技术不仅考虑到相关的跨项目工作历史记录 (例如外部库体验), 还考虑开发人员在与拉要求确定她作为潜在的代码审阅者的专业知识。我们首先使用对 10 个商业项目和 10 个相关库的探索性研究来激励我们的技术。使用来自 10 个商业项目和 6 个开源项目的 17 115 请求的实验表明, 我们的技术提供了 85%-92% 的推荐精度, 约 86% 的精度和 79%-81% 的代码审阅者推荐召回率很高承诺。与最先进的技术相比, 也验证了我们的推荐技术的经验发现和优越性。少

2018 年 7 月 9 日提交;最初宣布 2018 年 7 月。

评论:第 38 届软件工程国际会议 (附录卷) (icse 2016), 第 222 页-231 页, 美国德克萨斯州奥斯汀, 2016 年 5 月

日记本参考:proc. icse-c 2016, 第 222 页--231

8. **第 1807. 01853[pdf,其他] cse**

多伊 10.114/2597073.2597121

对 github 拉请求的洞察

作者:mohammad masudur rahman, chanchal k. roy

摘要: 鉴于 github 项目中不成功的拉请求数量不断增加, 深入了解这些请求的成功和失败对于开发人员来说至关重要。在本文中, 我们提供了一个比较研究成功和不成功的拉请求 78 个 github 基地项目的 20 142 个开发人员从 103, 192 个分叉项目。在研究中, 我们分析拉请求讨论文本、项目特定信息 (例如域、成熟度) 和开发人员特定信息 (例如, 经验), 以便报告有用的见解, 并使用它们来对比成功的和成功的不成功的拉请求。我们相信, 我们的研究将帮助开发人员克服 github 中的拉请求问题, 并通过明智的决策来帮助项目管理员。少

2018 年 7 月 5 日提交;最初宣布 2018 年 7 月。

评论:第 11 届采矿软件库工作会议 (msr 2014), 第 336-367 页, 印度海得拉巴, 2014 年 5 月

9. **第 xiv: 1806. 09774[pdf,其他] cse**

多伊 10.1109/TSE.2018.2822270

静态和动态测试用例优先化技术如何在现代软件系统上执行? github 项目的广泛研究

作者:齐罗,凯文·莫兰,张玲明,丹尼斯·波希万克

摘要: 测试用例优先级 (tcp) 是一种日益重要的回归测试技术, 用于根据预定义的目标重新排序测试用例, 尤其是在敏捷实践获得采用的情况下。为了更好地理解这些技术, 我们进行了第一次广泛的研究, 旨在对四种静态 tcp 技术进行经验评估, 并将它们与跨几个质量指标的研究状态动态 tcp 技术进行比较。这项研究是在 58 个真实的 java 程序中进行的, 其中包括 714 kloc, 并得出了几个值得注意的观察结果。首先, 我们在两个有效性指标 (检测到 apfd 的故障的平均百分比和对 apfdc 的成本认识) 中的结果表明, 在测试类粒度时, 这些指标往往是相关的, 但这种相关性在测试方法中并不成立粒 度。其次, 我们的分析表明, 静态技术可以是惊人的有效, 特别是当测量由 apfdc。第三, 我们发现 tcp 技术在较大的程序上的性能更高, 但程序大小不会影响技术之间的比较性能度量。第四, 软件的进化不会显著影响 tcp 技术之间的比较性能结果。第五, 在典型的实验环境下, 突变体的数量和类型都没有显著地影响 tcp 的有效性。最后, 我们的相似性分析表明, 高度优先的测试用例往往会发现不同的故障。少

2018 年 6 月 25 日提交;最初宣布 2018 年 6 月。

评论:预印接受的论文到 ieee 软件工程交易

日记本参考:问: 罗、莫兰、张大和波希万克, "静态和动态测试用例优先化技术在现代软件系统上如何执行? 关于 github 项目的广泛研究, "在 ieee 软件工程交易中, 2018 年

10. **第 xiv: 1806.08457[pdf,其他] cse**

您要呼叫谁? : github 讨论中 @-菜单的决定因素

作者:david kavaler, premkumar devanbu, vladimir filkov

摘要: 开源软件 (oss) 项目的成功取决于人群的贡献。当基于拉请求的系统中出现问题时, @ 提及被用来号召人们进行任务;以前的研究表明, 讨论中的 @ 提及与更快的问题解决方法有关。在大多数项目中, 可能有许多开发人员在技术上能够处理各种任务。但 oss 支持分布在广泛的社会和地理背景以及参与程度的动态团队。那么, 重要的是要知道该找谁, 即谁可以被依赖或信任承担与任务有关的重要职责, 以及原因。在本文中, 我们试图了解开发人员的哪些可观察到的社会技术属性可以用来构建一个很好的模型, 即它们在 github 问题中被提及, 并拉请求讨论。我们构建了未来 @ 提及的总体和项目特定的预测模型, 以便在 200 个 github 项目中的每一个项目中捕获 @ 提及的决定因素, 并了解这些决定因素在不同项目之间是否以及如何不同。我们发现, 可见性、专业知识和生产率与 @ 提及的增加有关, 而在许多控制变量的存在下, 响应能力并不高。此外, 我们发现, 虽然存在特定于项目的差异, 但整个模型可用于跨项目预测, 这表明了其 github 范围的实用程序。少

2018 年 6 月 21 日提交;最初宣布 2018 年 6 月。

评论:12 页, 5 个数字, 2 个表

类:D.2。2

11. 第 1804. 04749[pdf,其他] Cs。Cl

github 和堆栈溢出集合主题建模的每语料库配置

作者:christoph treude, markus wagner

摘要: 为了理解大量的文本数据, 主题建模经常被用作文本挖掘工具, 用于发现文本正文中隐藏的语义结构。潜在的 dirichlet 分配 (lda) 是一种常用的主题模型, 旨在通过对文本进行分组来解释语料库的结构。lda 需要多个参数才能正常工作, 在如何设置这些参数方面只有粗略的、有时相互冲突的准则。在本文中, 我们贡献 (一) 广泛的参数研究, 以获得良好的局部优化; (二) 一个后验的描述文本语料库相关的八种编程语言从 github 和堆栈溢出; (iii) 语料体特征的分析通过每个语料库 lda 配置的重要性。少

2018 年 6 月 23 日提交;v1 于 2018 年 4 月 12 日提交;最初宣布 2018 年 4 月。

12. 第 xiv: 1803.07764[pdf,其他] cse

估计源代码的缺陷: 一种基于 github 内容的预测模型

作者:ritu kapur, balwinder sodhi

摘要: 本文提出的两个主要贡献是: i) 一种方法, 用于构建包含从开源软件 (oss) 和相关错误报告中提取的源文件中提取的源代码特征的数据集, 二) 估计缺陷的预测模型给定的源代码。这些工件可用于构建与多个自动化软件工程领域相关的工具和技术, 如错误本地化、代码评审、建议和程序修复。为了实现我们的目标, 我们首先提取 github 上存在的源代码文件的编码样式信息 (例如与源代码中使用的编程语言构造相关的信息)。然后提取与这些源代码文件关联的 bug 报告 (如果有) 中可用的信息。这样提取的非结构化信息就会被转换为结构化知识库。我们审议了来自 20 个不同 github 存储库的 30400 多个源代码文件, 其中包括 4 个 bug 跟踪门户中大约 14950 个相关的 bug 报告。所考虑的源代码文件是用四种编程语言 (即、c、c++、java 和 python) 编写的, 属于不同类型的应用程序。然后使用知识库训练用于估计给定输入源代码的缺陷的机器学习 (ml) 模型。为了选择最佳的 ml 模型, 我们评估了 8 种不同的 ml 算法, 如随机林、k 最近邻居和支持向量机, 它们具有大约 50 个参数配置, 以比较它们在我们任务中的性能。我们的研究结果之一表明, 最好的 k 折叠 (k=5) 交叉验证结果是通过 nusvm 技术, 给出了 f1 的平均分数 0.914。少

2018 年 3 月 21 日提交;最初宣布 2018 年 3 月。

评论:提交给 2018 年。关键词: 维护软件;源代码挖掘;软件缺陷识别;自动化软件工程;软件工程中的人工智能

13. **建议: 1803.03175[pdf] cse**

多伊 10.18293/SEKE2018-085

大型开源生态系统公共发展项目的自动检测:基于 github 的探索性研究

作者:易成,李兵,李增阳,彭亮

摘要: github 托管了超过 1000 万个软件项目,是研究开发人员和软件项目行为的最重要数据源之一。但是,随着开源数据集大小的增加,挖掘这些数据集的潜在威胁也在增加。随着数据集的增长,人类确认所有样本的质量变得逐渐不现实。一些研究调查了这一问题,并提供了避免样本选择威胁的解决方案,但其中一些解决方案 (例如,寻找发展项目) 需要人为干预。当需要处理的数据量增加时,这些半自动解决方案就变得不那么有用了,因为需要人为干预的努力远远超出了负担得起的范围。为了解决这一问题,我们对 ghtorrent 数据集进行了调查,并提出了一种公共开发项目的检测方法。结果表明,该方法能通过两种方式有效地改进样品选择过程: (1) 提供一个简单的样品自动选择模型 (精度为 0.827,召回 0.827);(2) 我们还提供了一个复杂的模型,以帮助研究人员仔细筛选样品 (比手动确认所有样品的工作量减少了 63.2,并可实现 0.926 精度和 0.926 召回)。少

2018 年 5 月 8 日提交;v1 于 2018 年 3 月 8 日提交;最初宣布 2018 年 3 月。

评论:被 seke2018 会议接受

14. **建议: 1802. 08441[pdf,其他] cse**

多伊 10.114/3278142.32 278143

(否)持续集成对 github 项目提交活动的影响

作者:sebastian baltes, jascha knack, daniel anastasiou, ralf tymann, stepan diehl

摘要: 持续集成 (ci) 的核心目标是对软件项目进行小的增量更改,这些项目经常集成到主线存储库或分支中。本文提出了一项实证研究,研究在项目开始使用 ci 后,开发商是否调整了他们对上述目标的承诺活动。我们分析了 93 个 github 项目的提交和合并活动,这些项目引入了托管 ci 系统 travis ci,但此前在引入 ci 之前已经开发了至少一年。在我们的分析中,我们只发现了一个不可忽视的影响,即合并比率的增加,这意味着在项目开始使用 travis ci 之后,对所有提交有更多的合并提交。相关工作也报告了这一影响。然而,我们在 60 个 github 项目的随机抽样中观察到了同样的效果,这些项目没有使用 ci。因此,这种影响不可能仅仅是由于引入 ci 造成的。我们的结论是: (1) 在我们的项目样本中, ci 的引入并没有导致开发人员提交活动的重大变化, (2) 在将效果归因于可能不是原因的处理之前,将提交活动与基线进行比较是很重要的。或观察到的效果。少

2018 年 9 月 14 日提交;v1 于 2018 年 2 月 23 日提交;最初宣布 2018 年 2 月。

评论:7 页, 3 个数字, 第四届 sisoft 软件分析国际研讨会论文集 (swan 2018)

15. **建议: 1802. 06997[pdf,其他] cse**

对 github 自述文件的内容进行分类

作者:gede artha azriadi prana, christoph treude, ferdian thung, thushari atapattu, david lo

摘要: readme 文件在塑造开发人员对软件存储库的第一印象和记录存储库所承载的软件项目方面发挥着至关重要的作用。然而，我们缺乏对典型 readme 文件的内容以及可以自动处理这些文件的工具的系统理解。为了弥补这一差距，我们进行了一项定性研究，对来自 393 个随机采样的 github 存储库中的 4,226 自述自述文件部分进行了手动注释，并设计和评估了一个分类器和一组功能，可以对这些部分进行分类自动。我们发现讨论存储库的 "内容" 和 "如何" 的信息非常常见，而许多自述文件缺乏有关存储库的目的和状态的信息。我们的多标签分类器可以预测八个不同的类别，获得 f1 分数 0.746。为了评估分类的有效性，我们使用自动确定的类使用徽章标记 github readme 文件中的节，并向 20 名软件专业人员显示有或没有这些徽章的文件。大多数参与者都认为基于我们的分类器对部分进行了自动标记，以简化信息发现。这项工作使软件存储库的所有者能够提高其文档的质量，并有可能使软件开发社区更容易发现 github 自述文件中的相关信息。少

2018 年 7 月 30 日提交;v1 于 2018 年 2 月 20 日提交;最初宣布 2018 年 2 月。

16. **建议: 1802.0 2938[pdf,其他]** cse

堆栈溢出代码段在 github 项目中的使用与归因

作者:sebastian baltes, stephan diehl

摘要: 堆栈溢出 (so) 是软件开发人员最受欢迎的问答网站，提供了大量可复制的代码段。使用这些片段会引起维护和法律问题。so 的许可证 (cc bi-sa 3.0) 要求归属，即引用原始问题或答案，并要求派生的工作采用兼容的许可证。虽然关于 so 的代码段许可模型和所需的属性存在激烈的争论，但对于在没有适当归因的情况下从 so 复制代码段的程度却知之甚少。我们提出了一个大规模的实证研究的结果，分析了非平凡的 java 代码段的使用和归属从 so 答案在公共 github (gh) 项目。我们遵循三种不同的方法对未归因用法的比率进行三角估计，并与软件开发人员进行了两次在线调查，以补充我们的结果。对于我们分析的不同项目集，包含提及 so 的文件的项目的比例在 3.3% 和 11.9 之间。我们发现，在所有包含 so 代码的分析存储库中，最多 1.8% 的存储库以与 cc by-sa 3.0 兼容的方式使用该代码。此外，我们估计，从 so 复制的代码段中，最多四分之一是按要求进行的。在接受调查的开发人员中，近一半的人承认在没有属性的情况下从 so 复制代码，约三分之二的人不知道 so 代码段的许可及其含义。少

2018 年 9 月 21 日提交;v1 于 2018 年 2 月 8 日提交;最初宣布 2018 年 2 月。

评论:44 页, 8 位数字，经验软件工程 (斯普林格)

17. **第: 1710.10427[pdf]** si

devrank: github 中的矿业影响开发者

作者:廖志芳,金浩志,李一凡, 赵本红,吴金松, 刘胜宗

摘要: 随着社会编码越来越流行，了解开发者的影响可以使各种应用受益，比如新项目和创新的广告。然而，现有的大部分作品只注重对非加权和齐质网络中的有影响力节点进行排名，而这些节点无法将适当的重要分数转移到真正的重要节点上。为了在 github 中对开发人员进行排名，我们定义了开发人员对吸引注意力的能力的影响，这可以用未来获得的关注者数量来衡量。我们进一步定义了一种新的方法 devrank，它通过影响传播来通过根据用户行为构建的异构网络对开发人员进行排名，包括 "提交" 和 "跟随"。实验结果表明, devrank 可以提高排序精度，并对 devrank 与其他链路分析算法的性能进行了比较。少

2017 年 10 月 28 日提交;最初宣布 2017 年 10 月。

18. **建议: 1710.003[pdf,其他] cse**

github 存储库中的异常事件

作者:christoph treude, larissa leite, maurício aniche

摘要: 在大型和活动的软件项目中，开发人员了解所有项目活动变得不切实际。虽然可能没有必要知道每一个提交或问题，但可以说，了解那些不寻常的承诺或问题很重要。为了调查这一假设，我们使用一个全面的项目可能不寻常的方法列表，在 200 个 github 项目中发现了异常事件，并要求 140 名负责或受这些事件影响的开发人员对这些事件的有用性发表评论。相应的信息。基于 2, 096 条答案，我们确定了开发人员认为特别有用的异常事件的子集，包括大量代码修改和异常数量的审查活动，以及关于这些答案背后原因的定性证据。我们的发现提供了一种方法，可以减少开发人员需要解析的信息量，以便及时了解其项目中的开发活动。少

2018 年 4 月 30 日提交;v1 于 2017 年 10 月 5 日提交;最初宣布 2017 年 10 月。

评论:可在《系统和软件杂志》上发表

19. **第: 1708. 06860[pdf,其他] si**

github 和堆栈溢出: 分析跨多个社会协作平台的开发人员兴趣

作者:李嘉伟,卢国荣

摘要: 越来越多的软件开发人员正在使用广泛的社交协作平台进行软件开发和学习。在这项工作中，我们研究了 github 和堆栈溢出内部和之间开发人员兴趣的相似之处。我们的研究发现，开发人员在 github 和 stack 溢出中的共同利益平均为 39% 的 github 存储库和堆栈溢出问题，而开发人员参与的问题与共同利益一致。另外，开发人员确实与在这两个平台上共同参与活动的其他开发人员有着类似的兴趣。特别是，共同提交和共同请求相同 github 存储库并共同回答相同堆栈溢出问题的开发人员与其他共同参与其他平台活动的开发人员相比，具有更多的共同兴趣。少

2017 年 8 月 24 日提交;v1 于 2017 年 8 月 22 日提交;最初宣布 2017 年 8 月。

评论:第九届国际社会信息学会议预印 (socinfo 2017)

类:H.2 。8

20. **第 07.700452[pdf,其他] cse**

多伊 10.1109/ICSE-C.2017.99

需要归因: github 项目中的堆栈溢出代码段

作者:sebastian baltes, richard kiefer, stepan diehl

摘要: 堆栈溢出 (so) 是开发人员最大的问答网站，提供了大量可复制的代码段。使用这些片段会引起各种维护和法律问题。so 许可证要求归属，即引用原始问题或答案，并要求派生的工作采用兼容的许可证。虽然关于 so 的代码段许可模型和所需的属性存在激烈的争论，但对于在没有适当归因的情况下从 so 复制代码段的程度却知之甚少。本文介绍了 github 项目中 so 代码段的研究设计，并总结了实证研究的结果。平均而言, 3.22 的所有分析存储库和 7.33 的流行存储库包含了对 so 的引用。此外，我们发现开发人员更确切地说，是指 so 上的整个线程，而不是一个特定的答案。对于 java，至少有三分之二的复制代码段没有被归因。少

2017 年 7 月 3 日提交;最初宣布 2017 年 7 月。

评论:3 页, 1 个图, 第 39 届软件工程陪同国际会议论文集 (icse-c 2017), ieee, 2017, 161-163 页

21. **第 xiv:170 6.0 02777[pdf,其他] cs. cy**

多伊 10.17605/osf。io-enrq5

2017 年 github 开源调查摘要分析

作者:r. stuart geiger

摘要:本报告是对 2017 年 github 开源调查数据集的高级摘要分析, 介绍了调查中提出的每个问题的频率计数、比例、频率或比例条形图.

提交至 2017 年 6 月 8 日;最初宣布 2017 年 6 月。

评论:58 页

22. **第 07iv:170 5.001198[pdf,其他] cse**

吉苏布的堆栈溢出: 有什么片段吗?

作者:di yang, pedro martins, vaibhav sani, cristina lopes

摘要: 当程序员寻找如何实现某些编程任务时, 堆栈溢出是搜索引擎结果中的一个热门目的地。多年来, stack 溢出积累了一个令人印象深刻的代码段知识库, 这些代码段已被充分记录。我们有兴趣研究程序员如何在他们的项目中使用这些代码段。我们能在实际项目中找到堆栈溢出代码段吗? 使用代码段时, 此副本是文本还是经过调整? 而这些适应专业化是由目标工件的特性所要求的, 还是由程序员的特定要求所驱动? 本文上介绍的大规模研究分析了在 github 上托管的 909k 非叉 python 项目, 其中包含 290m 函数定义和在堆栈溢出中捕获的 190 万 python 片段。结果是对块级代码克隆内部和堆栈内部溢出和 github 的定量分析, 以及对我们的研究结果进行定性分析的编程行为分析。少

2017 年 5 月 2 日提交;最初宣布 2017 年 5 月。

评论:第十四届采矿软件存储处国际会议, 11 页

23. **建议: 1702.08571[pdf,其他] cse**

使用众包复制和扩展定性分析: 基于 github 的案例研究

作者:di chen, kathryn t. stolee, tim menzies

摘要: 由于在复制和扩大定性研究方面的困难, 这类研究很少得到核实。因此, 在本文中, 我们利用众包的优势 (低成本、快速、可扩展的劳动力) 来复制和扩大一项最先进的定性研究。该定性研究探讨了 20 个 github 拉请求, 以了解影响在审批和合并方面的拉力请求命运的因素。作为一项二级研究, 我们以 200 美元的成本使用众包, 研究了 142 个 github 项目的 250 个拉请求。之前的定性发现被映射到人群工人的问题。他们的答案被转换为二进制功能, 以建立一个预测, 预测代码是否会与 f1 分数中值为 68% 的合并。对于同一组拉请求, f1 中值分数可以通过使用先前定量结果定义的附加功能构建的预测值实现 90%。在本案例研究的基础上, 我们得出结论, 结合不同的研究方法有很大的好处。虽然定性洞察对于发现新的见解非常有用, 但它们很难扩展或复制。尽管如此, 他们可以指导和定义可扩展的二级研究的目标, 这些研究使用 (例如) 众包 + 数据挖掘。另一方面, 虽然数据挖掘方法是可重现的, 并且可扩展到大型数据集, 但它们的结果可能是非常错误的, 因为它们缺乏上下文信息。尽管如此, 它们可以用来检验从定性分析中获得的见解的稳定性和外部有效性。少

2017 年 3 月 1 日提交;v1 于 2017 年 2 月 27 日提交;最初宣布 2017 年 2 月。

评论:提交至 fsea17, 12 页

24. **特别报告: 1607. 04342[pdf,其他] cse**

多伊 10.114/297295982972929 66

预测 github 存储库的流行程度

作者:hudson borges, andre hora, marco tulio valente

摘要: github 是世界上最大的源代码存储库。它提供了一个基于信息的源代码管理平台，也提供了许多受社交网络启发的功能。例如, github 用户可以通过向项目添加星星来表示赞赏。因此，存储库的明星数量是衡量其受欢迎程度的直接标准。在本文中，我们使用多个线性回归来预测 github 存储库的星数。这些预测对存储库所有者和客户都很有用，他们通常想知道自己的项目在竞争激烈的开源开发市场中的表现。在一次大规模的分析中，我们显示，在接受了过去六个月收到的恒星数量的培训后，拟议的模型开始提供准确的预测。此外，对于增长缓慢的存储库和/或具有较少星级的存储库，建议使用具有相同增长趋势的存储库生成特定模型。最后，我们评估预测的能力不是预测存储库的星数，而是预测它在 github 存储库中的排名。我们发现预测和实际排名之间有非常强的相关性 (斯皮尔曼的 rho 大于 0.95)。少

2016 年 7 月 14 日提交;最初宣布 2016 年 7 月。

评论:第 12 届软件工程预测模型和数据分析国际会议 (承诺) 接受的论文预印

25. **特别报告: 1607. 02459[pdf,其他] cse**

多伊 10.115/290292929 50305

为什么我们要重构? github 贡献者的自白

作者:达尼洛·席尔瓦, nivilaos tsantalis, marco tulio valente

摘要: 重构是一种普遍的做法，可帮助开发人员提高其代码的可维护性和可读性。但是，针对开发人员应用的特定重构操作背后的实际动机进行的经验性研究数量有限。为了填补这一空白，我们监视了 github 上承载的 java 项目，以检测最近应用的重构，并要求开发人员找出他们决定重构代码的原因。通过对收集到的响应进行专题分析，我们为 12 种已知的重构类型编制了一份 44 种不同动机的目录。我们发现重构活动主要是由需求的变化驱动的，而更不是由代码气味驱动的。提取方法是最通用的重构操作，可用于 11 种不同的用途。最后，我们发现了开发人员使用的 ide 影响自动重构工具的采用的证据。少

2016 年 7 月 8 日提交;最初宣布 2016 年 7 月。

评论:在第 24 届软件工程基础国际研讨会 (fse) 上接受论文，第 1-12 页, 2016 年

26. **特别报告: 1606.05431[pdf, ps,其他] cse**

匿名电子邮件采访在 racan 和 github 上活跃的 r 包维护者

作者:汤姆·门斯

摘要: 本技术报告附有一篇研究文章，该研究文章对统计计算的 r 生态系统中的存储库包依赖关系相关的问题进行了经验性研究，重点是在 racan 和 github 上托管的 r 包。本报告提供补充材料，复制了 2015 年 11 月与 5 名活跃的 r 包维护人员进行的匿名和消毒的电子邮件访谈。目标是更好地了解 r 包维护人员如何通过 github 和 fran 开发和分发他们的包。所有 5 名受访者都在 github 上积极维持包裹，有的还活跃在 racan 上。他们是根据他们的个人资料 (他们在 github 和/或以往的 ran 上保留的 r 包的数量) 及其性别 (3 名受访者为男性, 2 名为女性) 选定的。少

2016 年 6 月 17 日提交;最初宣布 2016 年 6 月。

评论:5 页，无数字，技术报告

27. **第 xiv:1606.04984[pdf,其他] cse**

多伊 10.1109/ICSME.2016.31

了解影响 github 存储库普及程度的因素

作者:hudson borges, andre hora, marco tulio valente

摘要: 软件普及对现代开源开发者来说是一条有价值的信息，他们一直想知道他们的系统是否在吸引新用户，新版本是否正在被接受，或者他们是否满足了用户的期望。本文介绍了一项关于 github 托管软件系统受欢迎程度的研究, github 是世界上最大的开源软件集合。github 为用户提供了一种明确的方式来显示他们对托管存储库的满意度: 观星器按钮。在我们的研究中，我们揭示了影响 github 项目明星数量的主要因素，包括编程语言和应用领域。我们还研究了新功能对项目受欢迎程度的影响。最后，我们确定了四个主要的受欢迎增长模式，这些模式是在对代表 2,279 颗流行的 github 存储库的明星数量的时间序列进行分组后得出的。我们希望我们的结果为开发人员和维护人员提供有价值的见解，帮助他们在竞争激烈的软件市场中构建和发展系统。少

2016 年 7 月 14 日提交;v1 于 2016 年 6 月 15 日提交;最初宣布 2016 年 6 月。

评论:参加第 32 届软件维护和进化国际会议 (icsme 2016)。相机就绪版本

28. **特别报告: 1606.00521[pdf,其他] cse**

github 持续集成的初始和最终软件质量

作者:yue yu, bogdan vasilescu, huaimin wang, vladimir filkov, premkumar devanbu

摘要: 对新功能和错误修复的不断需求迫使软件项目缩短周期并更快地交付更新，同时保持软件质量。通过按需实现持续集成 (ci)，廉价的虚拟化云计算帮助缩短了计划。像 github 这样的平台支持云中的 ci。在使用 ci 的项目中，提交请求的用户将触发 ci 步骤。除了加快构建和测试，这还偶然创建了大量的构建和测试成功和失败的存档。ci 是一个相对较新的现象，这些档案可以对 ci 进行详细的研究。暴露了多少个问题? 它们发生在哪里? 影响 ci 故障的因素是什么? ci 确定的 "初始质量" 是否预测代码中以后会出现多少错误 ("最终质量")? 在本文中，我们对这些记录进行了大规模、精细的分辨率研究，以更好地了解 ci 过程、ci 故障的性质和预测因素，以及 ci 故障与代码最终质量的关系。我们发现: a) ci 故障似乎集中在几个文件中，就像正常的错误一样;b) ci 故障与最终故障的相关性不是很高;c) 在请求中使用 ci 不一定意味着该请求中的代码质量良好。少

2016 年 6 月 1 日提交;最初宣布 2016 年 6 月。

29. **第 xiv:1603.00431[pdf,其他] cs.PL**

试论 github 的编程语言

作者:amirali sanatinia, guevara noubir

文摘: github 是应用最广泛的社交分布式版本控制系统。它拥有约 1 000 万注册用户，拥有 1 600 多万公共存储库。其用户群也非常活跃, github 在前 100 名亚历克莎最受欢迎的网站中名列前茅。在本研究中，我们收集了 github 的完整状态。这样做，使我们能够研究生态系统的新方面。尽管 github 是数百万用户和存储库的所在地，但对用户活动时间序列的分析显示，只有约 10% 的用户可以被视为活动。收集的数据集使我们能够调查编程语言的普及程度以及在用户、存储库和编程语言之间的关系中存在的

patterns。通过将 k 均值聚类方法应用到用户-库提交矩阵中，我们发现两个清晰的编程语言集群与其余的编程语言是分开的。一个用于 "web 编程" 语言 (java 脚本、ruby、php、css) 的群集形式，另一个用于 "面向系统编程" 语言 (c、c++、python) 的群集形式。进一步的分类，让我们在 github 中构建一个使用编程语言的系统发育树。此外，我们还更详细地研究了前 1000 个存储库的主要和辅助编程语言。我们使用各种指标 (如代码行的百分比和年龄排名) 提供这些辅助编程语言的排名。少

2016 年 3 月 1 日提交;最初宣布 2016 年 3 月。

30. 建议: 160002594[pdf,其他] si

github 开源项目推荐系统

作者:tadej matek, svit timej zebec

摘要: 软件项目的托管平台可以形成协作社交网络，这方面的一个主要例子是 github，它可以说是这类平台中最受欢迎的平台。开源项目推荐系统可以成为 github 这样的平台的一大功能，使其用户能够快速、简单地找到相关项目。我们对基于 github 数据的构造图进行了网络分析，并提出了一个使用链接预测的推荐系统。少

2016 年 2 月 8 日提交;最初宣布 2016 年 2 月。

31. 建议: 1601.07862[pdf,其他] cs. cy

西班牙 github 用户数量的演变

作者:jj merelo

摘要: 自从我们开始测量西班牙 github 用户群体以来，它的数量一直在增加，在不到一年的时间里，它已经增长了近 50%，达到目前的 12000。然而，其原因尚不清楚。在本文中，我们将尝试找出在这种增长中的不同组成部分，或至少那些可以测量，以找出哪些是由于测量本身，哪些可能是由于其他原因。在本文中，我们在找出这种增长的原因方面取得了进展。少

2016 年 2 月 4 日提交;v1 于 2016 年 1 月 28 日提交;最初宣布 2016 年 1 月。

评论:支持 flossmectrics 闪存会谈的报告第三个版本更新至 1 月底

报告编号:geneur-2016-1

32. 第 1512.05558[pdf,其他] cse

多伊 10.115/290292950319

跨越 github 的无参数概率 api 挖掘

作者:jaroslav ffkes, charles sutton

摘要: 现有的 api 挖掘算法可能难以使用，因为它们需要昂贵的参数调优，并且返回的 api 调用集可能很大，非常冗余且难以理解。为了解决这个问题，我们提出了 pam (概率 api 矿工)，这是一种用于挖掘最有趣的 api 调用模式的近参数概率算法。我们表明, pam 在从 github 检索相关 api 调用序列时，其性能明显优于 mapo 和 upminer，实现了 69% 的测试集精度。此外，我们重点关注开发人员明确提供代码示例的库，这些库从数据集中的 967 个客户端项目中生成了超过 300, 000 个链写 api 示例代码。此评估表明，手写示例实际上对实际 api 用法的覆盖率有限。少

2016 年 11 月 11 日提交;v1 于 2015 年 12 月 17 日提交;最初宣布 2015 年 12 月。

评论:fse 2016 中的 12 页: 2016 年第 24 届 sisoft 软件工程基础国际研讨会论文集

33. 第 xiv:1512.01862[pdf,其他] cse

多伊 10.1109/ICSME.2014.62

社交编码世界中的持续集成:来自 github 的经验证据。 ** 更新版本，并进行更正 **

作者:bogdan vasilescu, stef van schuylenburg, jules wulms,亚历山大 serebrenik, mark g. j. van den 品牌

摘要: 持续集成是一种软件工程实践，它经常将所有开发人员的工作副本与共享的主分支合并，例如，每天多次。随着 github 的出现，协作软件开发从未像现在这样突出，github 是一个以其 "社交编码" 功能而闻名的平台，它有助于协作和共享，目前是开源世界中最大的代码主机。在 github 开发中，可以区分两种类型的开发人员对项目的贡献：直接贡献，来自对主项目存储库具有写访问权限的典型开发人员组，另一种是来自分叉开发人员的间接贡献主存储库，在本地更新其副本，并提交审核和合并的拉请求。在本文中，github 开发人员如何使用持续集成，以及贡献类型 (直接与间接) 和不同的项目特征 (例如，主要编程语言或项目年龄) 是否与成功相关自动生成。少

2015 年 12 月 6 日提交;最初宣布 2015 年 12 月。

评论:这是我们的 icsme 2014 文件的更新和更正版本: http://dx.doi.org/10.1109/ICSME.2014.62

34. **第 xiv:1007 00604[pdf,其他]** cse

github 应用程序的受欢迎程度：一个初步的说明

作者:hudson borges, marco tulio valente, andre hora, jailton coelho

摘要: github 是世界上最大的开源软件集合。因此，对软件开发人员和用户来说，比较和跟踪 github 存储库的受欢迎程度都非常重要。在本文中，我们提出了一个框架，以评估 github 软件的受欢迎程度，使用他们的明星数量。我们还提出了一套流行增长模式，描述一个系统的恒星数量随时间的演变。我们表明，恒星倾向于与其他措施 (如叉子) 相关联，并与第三方程序有效使用 github 软件相联系。在整个过程中，我们使用从 github 中提取的真实数据来说明框架的应用。少

2017 年 3 月 21 日提交;v1 于 2015 年 7 月 2 日提交;最初宣布 2015 年 7 月。

评论:该论文的扩展和修订版本出现在 icsme 2017 上: hudson borges、andre hora、marco tulio valente。了解影响 github 存储库普及程度的因素。在第 32 届 ieee 软件维护和进化国际会议 (icsme) 中，第 334 页，2016 年。https://arxiv.org/abs/1606.04984

35. **修订: 1501.06857[pdf,其他]** si

测量本地 github 开发人员社区

作者:j. j. merelo , nuria rico, israel blancas, m. g.arenas, fernando tricas, joséantonio vacas

摘要: 创建排名似乎是一个徒劳的练习，在肚脐凝视，尤其是在人们与这种事情，如程序员。然而，在本文中，我们将试图证明在西班牙创建基于城市 (或省) 的排名是如何导致各种有趣的影响的，包括提高生产力和社区建设。我们描述了我们在寻找居住在特定省份的程序员时使用的方法，重点是那些大多数人口集中的人，并采取不同的措施来显示这些社区在结构、数量和生产力上的差异。少

2015 年 1 月 27 日提交;最初宣布 2015 年 1 月。

评论:在牙线社区计量会议上的论文支持演示

36. **第十四条: 1409.4253[pdf,其他]** cse

探索 github 项目开放性的三个指标

作者:valerio cosentino, javier luis canovas izquierdo, jordi cabot

摘要: 开源软件项目的发展得益于一群志愿者的帮助他们的发展。因此，这些项目的成功取决于它们吸引 (和留住) 开发者的能力。我们相信，一个项目的开放性，即新用户对它的积极贡献是多么容易，可以帮助使一个项目更具吸引力。为了探索软件项目的开放性，我们提出了三个指标，重点是: (1) 项目社区的分布, (2) 外部贡献的接受率, (3) 成为项目的官方合作者所需的时间。我们对这些指标进行了调整并将其应用于 github 项目的一个子集，从而对其开放性给出了一些实际的发现。少

2014 年 9 月 15 日提交;最初宣布 2014 年 9 月。

评论:4 页, 4 位数字，发送到采矿挑战轨道在 msr' 14，被拒绝

类:H.4.0;D.2.8;j 。4

37. 第 xiv: 1407. 2535[pdf,其他] si

大规模编码: 作为协作社交网络的 github

作者:antonio lima, luca rossi, mirco musolesi

摘要: github 是最受欢迎的开放源代码存储库。根据公司 2013 年 4 月的声明，它拥有 350 多万用户，截至 2013 年 12 月，拥有超过 1, 000 万个存储库。它有一个可公开访问的 api，自 2012 年 3 月以来，它还发布了在公共项目上发生的所有事件的流。github 用户之间的交互性质复杂，以不同的形式进行。开发人员创建和分叉存储库，推送代码，批准他人推送的代码，为他们喜爱的项目添加书签，并跟踪其他开发人员的活动。本文将 github 描述为一个社交网络和一个协作平台。据我们所知，这是首次对 github 上发生的相互作用进行定量研究。我们分析了该服务在 18 个月内 (2012 年 3 月 11 日至 2013 年 9 月 11 日) 的日志，描述了 1.384 万次事件，并获得了约 219 万用户和 568 万个存储库的信息，这两个存储库都是线性增长的。我们展示了每个项目的贡献者数量、每个项目的观察者和每个用户的关注者的分布显示出一种类似于权力法的形状。我们分析了社会关系和存储中介的协作模式，我们观察到社会关系的互惠水平非常低。我们还根据创作的事件来衡量每个用户的活动，我们观察到非常活跃的用户不一定有大量的关注者。最后，我们提供了活动中心的地理特征，并研究了距离如何影响协作。少

2014 年 7 月 9 日提交;最初宣布 2014 年 7 月。

评论:10 页, 12 个数字, 1 个表。第八届阿拉伯国际网络和社交媒体国际会议论文集 (icwsm 2014)