سیستم های بازبینی کد

زهرا کولیوند
دانشجوی ارشد مهندسی کامپیوتر نرم افزار دانشگاه پیام نور تهران
Zahra001524@gmail.com

چکیده

کشف راه‌حل‌های مناسب برای بازبینی کد (CRRS) و انتخاب مرورگر کد یکی از جنبه های مهم توسعه نرم افزار است و به عوامل مختلفی بستگی دارد. هنگام ایجاد نرم‌افزار باید در نظر گرفته شود و ابعاد مختلفی که بر اساس آنها می توان آنها را طبقه بندی کرد. هدف ما درک ویژگی های مهم CRRS و آنچه می‌تواند کیفیت بازرسی را بهبود ببخشد، است.مطالعه مروری بر ادبیات برای درک CRRS های موجود انجام شد. نظرسنجی از اعضای پروژه توسعه نرم افزار برای درک ویژگیهای مهم و مفقود شده در CRRS انجام شده است. مقالات را به دو دسته طبقه بندی کردیم : بر اساس نوع داده مورد استفاده برای ارائه توصیه ها و نوع پروژه مورد استفاده برای ارزیابی. این نظرسنجی به ما کمک کرد ویژگی های موجود در CRRS را درک کرده و برخی از روندها و الگوها را مشاهده کنیم. از شرکت کنندگان خواسته ایم تا برخی از ویژگی های غیر از ویژگی های ارائه شده را که هنگام انتخاب مرورگر کد به عنوان ویژگی مهم در نظر گرفته اند ، ارائه دهند.

واژگان کلیدی: بازبینی کد – برنامه نویسی جفتی- بررسی سیستماتیک

مقدمه

هدف بازبینی کد شناسایی و اصلاح اشتباهات در کد منبع و همچنین بهبود کیفیت کد و مهارت توسعه دهندگان نرم افزار است . علاوه بر هدف بهبود کیفیت کد یا یافتن نقص در کد منبع همچنین باعث هدف افزایش آگاهی تیم و کمک به توزیع دانش را نیز شامل می شود و مالکیت کد مشترک را تشویق می کند. چهار نوع کد وجود دارد:

1- برنامه نویسی جفتی: در این نوع بازبینی کد ، دو توسعه دهنده به طور همزمان کد منبع را تولید می کنند و به طور همزمان مرور می کنند.

2- مرور کد به کمک ابزار: برای این نوع مرور کد ، نویسندگان و توسعه دهندگان از ابزارهای بررسی کد همتا استفاده می کنند .

3- بازبینی کد مرور: در اینجا ، توسعه دهنده مرورگر را از طریق مجموعه ای از تغییرات کد راهنمایی می کند.

4- بازبینی رسمی کد: با دقت بسیار زیاد و طبق الگوها انجام می‌شود. بازرسی دقیق کد با مشارکت تعدادی از افراد متخصص و در چند مرحله انجام می‌شود.

مرور کد را می توان به عنوان بازرسی دستی تغییرات در کد منبع دانست. تعدادی ابزار و سیستم توصیه ای وجود دارد که به منظور بررسی کد توسط تعدادی از سازمانهای مختلف توسعه یافته است .

دلایل مختلفی وجود دارد که چرا علاوه بر یافتن عیوب کد ، به مرورگر کد نیز نیاز است. این امر به این دلیل است که مرورگران کد بر بهبود کد ، یافتن راه حل های جایگزین برای یک مشکل ، ارائه دانش ، بداهه پردازی در فرایند توسعه ، اجتناب از وقفه های ساختاری تمرکز می کنند.(Gusfiel,1997) ، اشتراک مالکیت کد ، و همچنین ارزیابی تیم (Bacchelli,2013). به عنوان مثال رحمان ، روی و کالینز(Rahman,2016) ، یک سیستم بازبینی کد را پیشنهاد کردند که در آن تخصص یک مرورگر کد بر اساس اطلاعات بدست آمده از یک پروژه متقابل است.

زمینه های متعددی وجود دارد که در آنها مشارکت سیستم های توصیه شده برای اعضای پروژه توسعه نرم افزار مفید بوده است. برای کمک به کار بررسی کد ، تحقیقات قابل توجهی در مورد سیستم های توصیه ای انجام شده است که هدف آنها ارائه توصیه های بازبینان کد بر اساس جنبه های مختلف است.

روش تحقیق

در این مقاله، از چند مقاله و چند منبع مرتبط با موضوع استفاده شده و روش انجام شده به صورت کتابخانه‌ای به صورت زیر است:

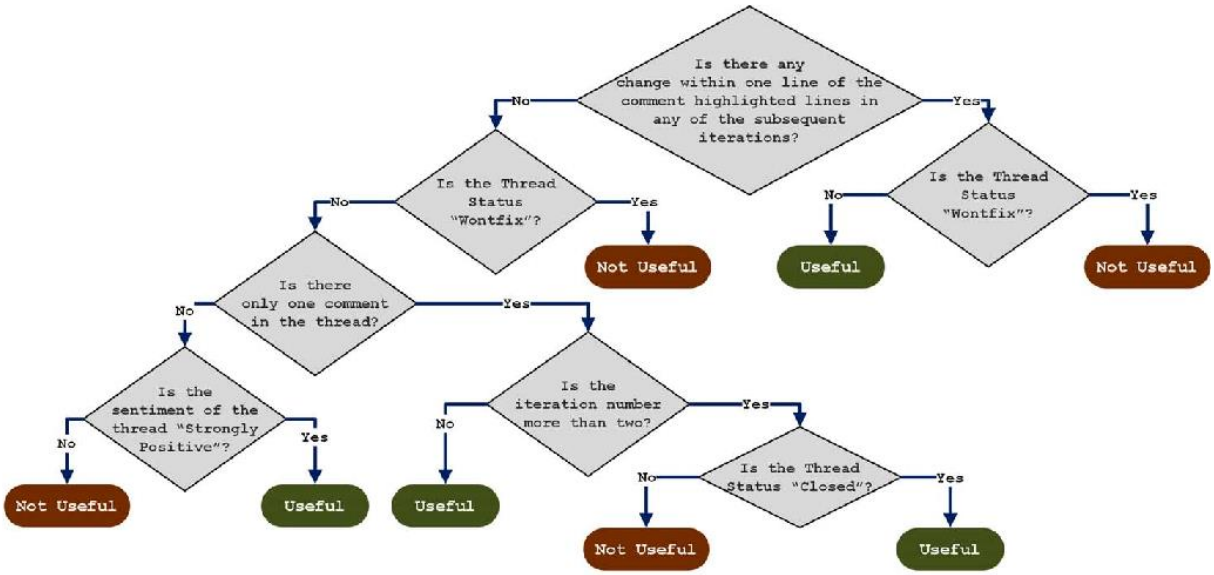
1. تعدادی از ویژگی های موجود در سیستم های پیشنهادی بازبینی کد (CRRS) را شناسایی کرده و آن ویژگی ها را بر اساس مفید بودن آنها
2. CRRS های موجود را بر اساس ابعاد مختلف طبقه بندی کردیم.
3. ویژگی هایی را که می توان هنگام انتخاب مرورگر کد مهم تلقی کرد ، شناسایی کردیم.
4. بهبودهای احتمالی CRRS های موجود را برای تسهیل یافتن مرورگران کد مناسب شناسایی کردیم.

یافته ها

مطالعه تحقیقاتی در سه مرحله انجام شد که در آن اولین گام به تشخیص نظرات بازبینی کد مفید و غیر مفید بر اساس مصاحبه با توسعه دهندگان کمک کرد. مصاحبه فردی نیمه ساختار یافته از توسعه دهندگان داشتن سطوح مختلف تجربه در

بررسی کد و توسعه کد از چهار مختلف مایکروسافت پروژه تی انجام شد. از مصاحبه شوندگان خواسته شد تا نظرات را از مقیاس 1- 3 (1- مفید ، 2- تا حدودی مفید و 3- مفید) امتیاز دهند. نتایج مصاحبه نشان داد که 69 comments از نظرات مرور "مفید" یا "تا حدودی مفید" بودند. توریم و نظرات بررسی است که نشان نقص عملکردی به عنوان در نظر گرفته شد مفید نظر. : از سوی دیگر، به نظر که به دسته بندی تعلق در کد مستندات ، نمایش تصویری از کد (به عنوان مثال خط خالی یا دندانه دار)، ارگان ization از کد (به عنوان مثال چگونه قابلیت به روش تقسیم) و رویکرد راه حل بود به عنوان در نظر گرفته تا حدودی مفید . همه نظراتی که یا مثبت کاذب بودند (به عنوان مثال به دلیل عدم تخصص هنگامی که داور مشکلی را در کد نشان می دهد) یا در هیچ طبقه ای قرار نگرفت همانطور که قبلاً ذکر شد به عنوان نظرات غیر مفید طبقه بندی شدند.

در مرحله دوم ، نویسندگان یک طبقه بندی خودکار با استفاده از یافته های به دست آمده از مرحله اول ایجاد کردند. به منظور ایجاد کلاس بهتر ، نویسندگان نظرات بازبینی را به صورت دستی به دو دسته مفید و غیر مفید طبقه بندی کردند . نظراتی که در مطالعه اکتشافی تا حدی مفید طبقه بندی شده اند ، در این مرحله دوم در دسته مفید قرار گرفتند. بر اساس مصاحبه و تجزیه و تحلیل دستی ، 8 ویژگی بعدی نظرات مشخص شد. بر اساس این ویژگی ها و دسته ها ، "مدل درخت تصمیم گیری برای طبقه بندی نظرات مفید" مطابق شکل 1 ساخته شد.



شکل 1- مدل درخت تصمیم‌گیری

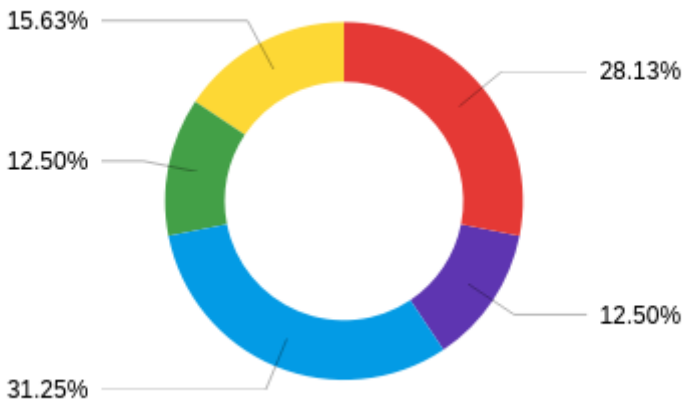
ما 30 پرس و جو در مورد نظرسنجی خود به داشتیم ، اما تنها 18 مورد از آنها به جلو حرکت کردند و به بررسی غربالگری ما پاسخ دادند. از بین این 15 پاسخ ، ما 4 پاسخ را فیلتر کردیم که با استفاده از نظرسنجی غربالگری حداقل معیارها را نداشتند . در زیر نتایج بدست آمده برای نظر سنجی سیستم ها و ابزارهای توصیف مرورگر جمعیت شناختی و کد ارائه شده است .

1. نقش شغلی شرکت کنندگان

بر اساس نتایج به دست آمده ، مشخص شد که اکثر شرکت کنندگان یا توسعه دهندگان ، یا مهندسان نرم افزار یا برنامه نویس بودند.

به غیر از ویژگی های CRRS ، همچنین در مورد ویژگی های مختلف که می تواند برای CRRS مفید واقع شود ، با استفاده بیشتر از تجربه کاربری ، قابل دسترسی و راحت نشان داد .درصد افرادی که ویژگی های زیر را مفید دانسته اند به شرح زیر با توزیع نمودار پای زیر نشان داده شده است:

- رائه خط لوله ای که نشان می دهد پروژه در چه مرحله ای است یعنی ساخت ، آزمایش ، بازنویسی کد ، استقرار و غیره 31.25 :٪ از شرکت کنندگان این ویژگی رابط کاربری را مفید و راحت تر می دانند.
- وجود پنل کاربری برای هر فرد که داده های آماری کلیه اقدامات انجام شده را نشان می دهد (مانند تعداد تعهدات ، تعداد بررسی کد انجام شده ، تعداد خطاها/هشدارهای کد در پروژه فعلی و غیره) 28 درصد از شرکت کنندگان این دیدگاه برای جستجوی جزئیات و اقدامات انجام شده توسط هر فرد مفید بود.
- هنگام تغییر کد ، بحث کدگذاری جدید و قدیمی با کد 15 درصد از شرکت کنندگان این ویژگی را مفید می دانند تا روند بازبینی کد را برای تازه کار و توسعه دهنده آسان تر کند.
- انتقال کد با استفاده از طرح رنگی با نام برنامه نویس 12.5 درصد از شرکت کنندگان این ویژگی را مفید دانستند.
- گزینه ای برای انتخاب یک "شاخه/پرونده" خاص در یک پروژه برای حفظ گردش کار سیستماتیک و روش بازبینی کد سازماندهی شده 12.5 درصد از شرکت کنندگان با سازماندهی بیشتر فرآیند بازبینی کد ، این ویژگی را مفید دانستند. نتایج در شکل 2 نشان داده شده است.



شکل2 – انواع ویژگی های CRRS

بحث و نتیجه‌گیری

در این مقاله CRRS و انواع ویژگی‌های آن بررسی شد و یافته‌ها ذکر شد. مطالعه ادبیات سیستماتیک گسترده ای را انجام دادیم که نه تنها به سیستم‌های توصیه مرورگر کد بلکه به شیوه ها و رویه های بازبینی کد نیز می‌پردازد. ارائه تصویر بهتر در مورد نیازهای اعضای پروژه نرم افزار در مورد بررسی کد در ارتباط با استفاده از CRRS از مهمترین نتایج این پروژه می‌باشد.
در این قسمت پیشنهاداتی جهت ادامه کار داریم . ایجاد سیستمی که تمام جزئیات مرورگر در داشبورد سیستم قابل مشاهده باشد (یعنی تجربه کار ، تجربه فناوری ، تعداد بررسی کد انجام شده و ...)

منابع

D. Gusfield, *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. 1997.

A. Bacchelli and C. Bird, “Expectations, outcomes, and challenges of modern code review,” D. Notkin, B. H. C. Cheng, and K. Pohl, Eds., pp. 712–721, 2013. DOI:10.1109/ICSE.2013.6606617. [Online]. Available: https://doi.org/10.1109/ ICSE.2013.6606617.

M. M. Rahman, C. K. Roy, and J. A. Collins, “Correct: Code reviewer recommendation in github based on cross-project and technology experience,” L. K. Dillon, W. Visser, and L. Williams, Eds., pp. 222–231, 2016. DOI: 10.1145/2889160.2889244. [Online]. Available: https://doi.org/10.1145/2889160.2889244.

Code review systems

Zahra Kolivand

Payam Noor University

Zahra001524@gmail.com

Abstract

Discovering the right solutions for code review (CRRS) and choosing a code browser is one of the most important aspects of software development and depends on several factors. When creating software, different dimensions must be considered based on which they can be classified. Our goal is to understand the important features of CRRS and what can improve inspection quality. A literature review study was conducted to understand the existing CRRS. Surveys of members of the software development project have been conducted to understand important and missing features in CRRS. We categorized the articles into two categories: based on the type of data used to provide recommendations and the type of project used for evaluation. This survey helped us understand the features of CRRS and observe some of the trends and patterns. Participants were asked to provide some of the non-featured features that they considered important when choosing a code browser.

Keywords: code reviewer, Pair programming, systematic examination