UNIVERSIDADE FEDERAL DE CAMPINA GRANDE CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

HUGO ADDOBBATI GALVÃO

Um Estudo Comparativo entre Pull Requests com e sem Refatoramentos através da análise de Revisões de Código

CAMPINA GRANDE - PB

2021

HUGO ADDOBBATI GALVÃO

UM ESTUDO COMPARATIVO ENTRE PULL REQUESTS COM E SEM REFATORAMENTOS ATRAVÉS DA ANÁLISE DE REVISÕES DE CÓDIGO

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Tiago Lima Massoni

CAMPINA GRANDE - PB

2021

G182e Galvão, Hugo Addobbati.
Um estudo comparativo entre Pull Requests com e sem refatoramentos através da análise de revisões de códigos. / Hugo Addobbati Galvão. – 2021.

13 f.

Orientador: Prof. Dr. Tiago Lima Massoni.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Pull requests. 2. Revisão de códigos. 3. Refatoramento. 4. Comentário de revisão. I. Massoni, Tiago Lima. II. Título.

CDU:004(045)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

HUGO ADDOBBATI GALVÃO

UM ESTUDO COMPARATIVO ENTRE PULL REQUESTS COM E SEM REFATORAMENTOS ATRAVÉS DA ANÁLISE DE REVISÕES DE CÓDIGO

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Tiago Lima Massoni

Orientador – UASC/CEEI/UFCG

Professor Dr. Jorge Cesar Abrantes de Figueiredo

Examinador – UASC/CEEI/UFCG

Professor Dr. Tiago Lima Massoni

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 20 de Outubro de 2021.

CAMPINA GRANDE - PB

ABSTRACT

Over the years, code review has changed; before, a manual inspection was carried out (strict and synchronous), whereas nowadays, a more modern revision (asynchronous and less rigorous) is carried out. Currently, Git, through the Github platform, is the most popular version control system, favoring several discussions related to changes in source code. With the help of tools like RefactoringMiner, which provides detection of refactorings applied to source codes and, using a sample of repositories from the Apache project on Github, this work, through manual inspections of review comments, aims to understand and characterize the comments that induced refactorings in the PRs, in order to understand the characteristics and differences of the review comments in PRs with and without refactorings. Through the hypotheses raised, we tried to complement the understanding of the qualitative part of the review comments, previously addressed in a similar way in another research, which analyzed qualitative and quantitative data from PRs that induced refactorings and PRs that did not induce refactorings, with the intention of understanding better the differences between the two types of PRs, at the Pull Request level.

Um Estudo Comparativo entre Pull Requests com e sem Refatoramentos através da análise de Revisões de Código

Hugo Addobbati Galvão
Universidade Federal de Campina Grande
hugo.galvao@ccc.ufcg.edu.br

Tiago Lima Massoni
Universidade Federal de Campina Grande
massoni@dsc.ufcg.edu.br

RESUMO

Com o passar dos anos, a revisão de código vem mudando; antes, era feita uma inspeção manual (rigorosa e síncrona), já nos dias atuais, é feita uma revisão mais moderna (assíncrona e menos rigorosa). Atualmente, o Git, através da plataforma Github, é o sistema de controle de versões mais popular, favorecendo diversas discussões relacionadas a mudanças no código-fonte. Com o auxílio de ferramentas como *RefactoringMiner*, que fornece a detecção de refatoramentos aplicados aos códigos-fonte e, utilizando-se de uma amostra de repositórios provenientes do projeto do Apache no Github, este trabalho, através de inspeções manuais de comentários de revisão, visa entender e caracterizar os comentários que induziram refatoramentos nos PRs, com o intuito de entender as características próprias e diferenças dos comentários de revisão em PRs com e sem refatoramentos. Através das hipóteses levantadas, tentamos complementar o entendimento da parte qualitativa dos comentários de revisão, abordados anteriormente de forma similar em outra pesquisa, que analisava dados qualitativos e quantitativos de PRs que induziram refatoramentos e de PRs que não induziram refatoramentos, com a intenção de entender melhor as diferenças entre os dois tipos de PRs, no nível de Pull Request.

Keywords

Pull Requests, Refatoramento, Git, Comentário de Revisão, Revisão de Código

1. INTRODUÇÃO

Existem diversas variáveis que influenciam o desenvolvimento e revisão de código nos dias modernos. Plataformas como o Github[2], que utilizam Git[3] como sistema de controle de versão, normalmente, costumam disponibilizar diversos recursos (discussões por comentário, visualização de histórico etc), além do código-fonte. Parte do entendimento das mudanças provém de discussões que ocorrem dentro de uma das etapas do processo de versionamento, o *Pull Request* (PR).

Os PRs permitem que outras pessoas visualizem mudanças no código e compartilhem conhecimento e sugestões sobre o que deve ser melhorado, como deve ser melhorado e porque deve ser melhorado. Nos PRs podem ocorrer refatoramentos, que, segundo a definição, são mudanças feitas dentro de um código, que alteraram sua estrutura interna sem alterar o comportamento, com o intuito de prover um código mais legível e limpo[3;8].

Esses refatoramentos, por sua vez, podem ter sido motivados por comentários de revisão ou discussão, assim, consideramos os PRs que induzem refatoramento, que são PRs em que ocorreram refatoramentos nos commits subsequentes à abertura do PR, e os PRs que não induzem refatoramento, que são PRs em que não ocorreram refatoramentos nos commits subsequentes à abertura do PR.

Essa classificação possui o propósito de entender as diferenças em termos de revisão de código entre os dois tipos, através da caracterização dos comentários de revisão de ambos os tipos, por análises qualitativas (comentário de review explicitou diretamente ou indiretamente uma mudança) e quantitativas (tamanho do comentário, se houve resposta para o comentário etc).

Os dados das análises serão agregados para compor os resultados, que apresentará métricas e informações estatísticas e hipóteses sobre os comentários de revisão e suas características, com o propósito de aumentar o entendimento sobre o processo de revisão de código nos PRs.

O estudo contou com quatro etapas de coleta de dados e análises. chegou-se a esse número pois foi decidido que a pesquisa continuaria gerando novas etapas até que se atingisse a saturação dos dados, ou seja, nenhuma informação nova estaria sendo gerada com mais etapas.

As principais hipóteses levantadas são que a ocorrência de comentários e sugestões sobre refatoramentos geralmente levam os refatoramentos a ocorrerem, porém a ausência de comentários também não faz com que os refatoramentos ocorram. É possível observar também que sugestões de mudanças de lógica estiveram presentes também na maioria de ambos os tipos de Pull Request, isso pode indicar que o processo de revisão relacionado a refatoramentos geralmente vem acompanhado de sugestões sobre a lógica do código.

2. BACKGROUND

O Pull Request é um elemento importante para a revisão de código moderno. Através de Pull Requests, os desenvolvedores podem contribuir para o código fonte com mudanças. Antes das mudanças serem aplicadas, os Pull Requests normalmente passam por um processo de revisão das mudanças no nível de código, para só então serem aprovados ou reprovados. Caso o PR seja aprovado, as mudanças são aplicadas ao código fonte.

A revisão do código geralmente se dá por forma de comentário de revisão, na qual o revisor do código debate ou não possíveis sugestões de alterações no código. Nesse contexto de Pull Request conseguimos caracterizar os Pull Requests que induzem refatoramentos, que são Pull Requests no qual pelo menos uma das mudanças adicionadas ao código após a sua abertura foi algum tipo de refatoramento.

Para meios de simplificação, nesse artigo, os Pull Requests que induziram refatoramentos serão abordados como *RIPRs*, sigla que provém do termo *Refactoring Inducing Pull Request*, e os Pull Requests que não induziram refatoramentos serão abordados como *non-RIPRs*.

A figura 1 mostra um exemplo de comentário de revisão, que sugeriu a mudança de um tipo de variável, e embaixo, o autor do Pull Request afirma que irá aplicar as mudanças sugeridas. A figura 2 exemplifica o processo de criação de um Pull Request, desde a parte de clone, até a aprovação e *merge* do PR.

Quanto aos *RIPRs e non-RIPRs*, a figura 3 nos ajuda a entender onde exatamente ocorrem os *RIPRs*, caso alguma mudança de refatoramento ocorra em algum dos commits após a abertura do Pull Request dentro da mesma branch do Pull Request, então o Pull Request é considerado um *RIPR*, caso nenhuma das mudanças seja relacionada a refatoramentos, o Pull Request é considerado um *non-RIPR*.

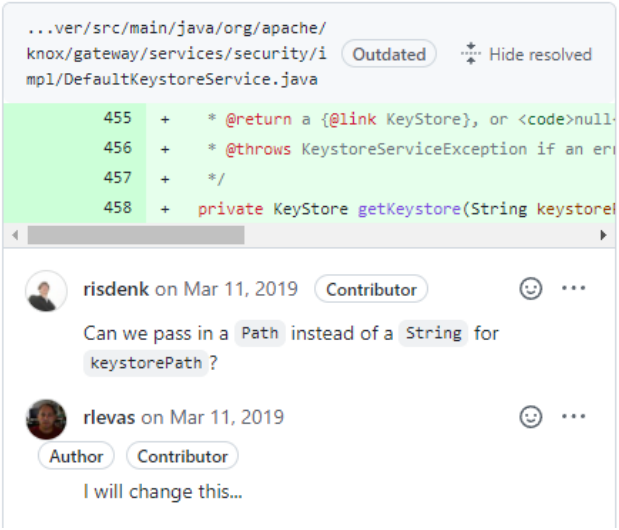

Figura 1: Exemplo de comentário de revisão em um Pull Request.

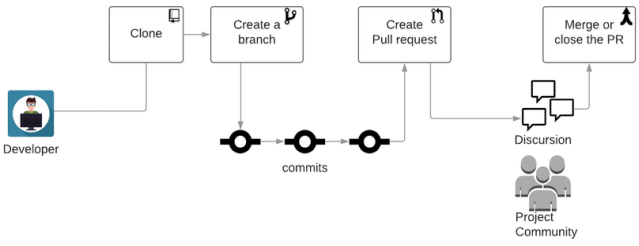

Figura 2: Visualização do processo de criação e revisão de um Pull Request[7].

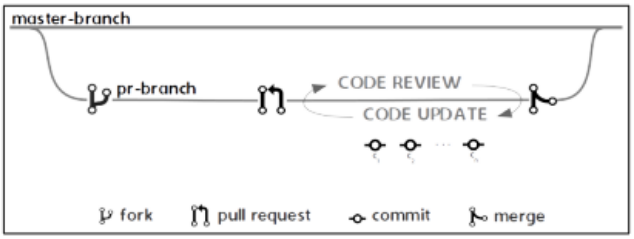

Figura 3: Exemplo de Fluxo de mudanças de código em um Pull Request.[1]

O tema da vigente pesquisa já foi abordado em uma pesquisa anterior[1], de forma que, essa pesquisa tenta complementar os resultados obtidos nessa pesquisa anterior. Enquanto essa pesquisa anterior trabalhou no nível de Pull Request, a vigente pesquisa trabalha estudando os Pull Requests no nível de comentário de revisão, tentando complementar o entendimento da parte qualitativa dos comentários.

A pesquisa anterior trouxe resultados que abrangiam dados quantitativos e qualitativos, que abrangiam desde características dos Pull Requests, como número médio de comentários e revisores, até números sobre quais os refatoramentos mais comuns em *RIPRs*.

Alguns dos principais dados e hipóteses interessantes levantados pela pesquisa anterior foram os seguintes:

- *RIPRs* são mais prováveis de possuir um número maior de linhas deletadas e linhas adicionadas do que *non-RIPRs.*
- *RIPRs* são mais prováveis de possuir um número maior de arquivos modificados do que *non-RIPRs.*
- *RIPRs* são mais prováveis de possuir um número maior de comentários de revisão do que *non-RIPRs.*
- *RIPRs* são mais prováveis de possuir um número maior de commits do que *non-RIPRs.*
- Não existem estatísticas que comprovem que *RIPRs* tenham diferenças em ter um número maior de revisores que *non-RIPRs.*
- 84,9% dos refatoramentos são realizados a partir do segundo commit após a abertura do PR.
- Os números médio e mediano de refatoramentos por Pull Request foram de 43,1 e 6 respectivamente.

3. METODOLOGIA

O estudo consistiu em estudar, entender, e analisar diferentes Pull Requests, retirados do repositório do Apache. O Apache foi escolhido devido à sua popularidade, pois ele possui mais de 350 projetos de código aberto, completamente migrados para o Github desde fevereiro de 2019, e o mesmo também possui mais de 7000 contribuidores de vários lugares distintos do mundo[4].

Para classificar os Pull Requests da amostra como Pull Requests que induziram refatoramentos ou Pull Requests que não induziram refatoramentos, foi utilizada a ferramenta *RefactoringMiner*, que é capaz de detectar a ocorrência de refatoramentos em código Java com uma precisão considerável de 99.6% e revocação de 94%[5;6]. Essa ferramenta tornou o estudo possível, pois dado o tamanho da amostra, a procura de refatoramentos no código tornaria a pesquisa certamente inviável.

A versão utilizada do *RefactoringMiner* foi a versão 1.0.0, que detecta até 40 tipos diferentes de refatoramentos ocorridos em um histórico de código Java.

3.1 Coleta de dados

O estudo coletou dados por etapas, utilizando-se de decisões empíricas baseadas nas análises anteriores para gerar amostras não probabilísticas a partir de uma amostra teórica de Pull Requests retirados do repositório do apache.

Conforme as análises foram sendo realizadas, novas coletas de dados foram sendo realizadas nas etapas subsequentes, levando em conta os dados e os resultados anteriormente obtidos para gerar insights sobre quais outros dados e características em Pull Requests seriam úteis para gerar mais resultados interessantes, conforme o trabalho foi avançando, as seguintes etapas de coletas de dado ocorreram:

3.1.1 Etapa 1

A etapa 1 considerou 10 *RIPRs* e 10 *non-RIPRs* que receberam 5 comentários de revisão de 2 revisores, aleatoriamente selecionados, a escolha dessa etapa inicial levou em consideração que seria importante começar analisando Pull Requests com um número médio de comentários e de revisores. Por isso, foram selecionados 5 comentários e 2 revisores, pois 5 é o número médio de comentários em Pull Requests da amostra e 2 é o número médio de revisores em Pull Requests da amostra.

3.1.2 Etapa 2

A etapa 2 considerou 10 *RIPRs* contendo apenas 1 refatoramento e apenas um commit subsequente, e 10 *non-RIPRs* com um commit subsequente. Essa configuração foi escolhida para ajudar a pesquisa a entender o cenário de Pull Requests simples e concisos.

3.1.3 Etapa 3

A etapa 3 considerou 13 *RIPRs* contemplando refactorings que mudam code design e 13 *non-RIPRs* aleatoriamente selecionados sem critério. Essa configuração foi escolhida para ajudar a pesquisa a entender se existe algum tipo de influência no tipo de refatoramento aplicado em um determinado Pull Request com os comentários do mesmo Pull Request.

3.1.4 Etapa 4

A etapa 4 considerou 13 *RIPRs* com uma sequência de pelo menos 2 refatoramentos que ocorreram em um mesmo commit, 13 *RIPRs* com uma sequência de pelo menos 2 refatoramentos que ocorreram em commits distintos e 26 *non-RIPRs* selecionados aleatoriamente sem critério. Essa configuração foi escolhida para ajudar a pesquisa a entender como a questão histórica pode influenciar os comentários de revisão.

3.2 Procedimento do estudo

Conforme mencionado anteriormente no tópico 3.1, o estudo começou gerando uma amostra teórica utilizando a ferramenta *RefactoringMiner*, que consegue identificar refatoramentos em qualquer parte de um histórico de código Java, com isso, todos os Pull Requests da amostra foram classificados como *RIPRs* ou *non-RIPRs.*

Então, por formato de etapas, amostras não-probabilísticas de *RIPRs* e *non-RIPRs* com características semelhantes, foram geradas em cada etapa. A tomada de decisão de qual característica seria considerada para a etapa subsequente seria realizada de forma empírica baseado na análise da amostra anterior.

O procedimento de análises e geração de novas análises se repetiu até a saturação dos dados, ou seja, a partir do momento que com repetidas análises nenhuma conclusão ou informação nova era obtida. As análises serão feitas buscando entender duas perguntas de pesquisa, sendo elas as seguintes:

Q1: Como os comentários dos PRs que induzem refatoramentos e dos PRs que não induzem refatoramentos são caracterizados?
Q2: Quais as diferenças entre os PRs que induzem refatoramentos e os PRs que não induzem refatoramentos, em termos de comentários de revisão?

Essas perguntas devem ser entendidas como de viés qualitativo, visto que esse trabalho busca hipóteses acerca da parte qualitativa dos comentários de revisão.

3.3 Método de análise

A análise realizada sobre as amostras de cada etapa era manual e empírica, vale salientar que a análise foi feita por dois revisores diferentes, devido ao caráter empírico da pesquisa, foi-se utilizado de uma equipe de dois pesquisadores para realizar a análise das amostras de cada etapa, tudo isso com o intuito de reduzir o possível viés causado por uma análise manual e empírica. A análise foi dividida de forma igual para ambos os pesquisadores, ou seja, ambos os pesquisadores analisaram todos os PRs da amostra, para assim, evitar o viés da pesquisa, através de diferentes visões e interpretações dos mesmos dados.

4. RESULTADOS

No final das análises manuais sobre todos os Pull Requests das 4 amostras explicadas na seção anterior, foram levantados alguns dados e hipóteses acerca das características dos Pull Requests, no nível de comentários de revisão, os dados são mostrados na Tabela 1.

Tabela 1: Ocorrência de cenários de comentários de revisão nos diferentes tipos de Pull Request analisados

Cenário possível de comentário de revisão	Contexto	*RIPRs*	*Non-RIPRs*
Pull Request teve comentário sugerindo mudança de nome.	*Com Justificativa*	11/58 (18,9%)	1/59 (1,6%)
	Sem Justificativa	11/58 (18,9%)	0/59 (0%)
	Total	22/58 (37,9%)	1/59 (1,6%)

Pull Request não teve nenhum comentário sugerindo alguma mudança relacionada à refatoramentos	*Total*	21/58 (36,2%)	58/59 (98,3%)
Pull Request teve comentários sugerindo refatoramentos (excluindo sugestões de mudança de nome).	*Total*	20/58 (34,4%)	0/59 (0%)
Pull Request teve comentários sugerindo mudanças relacionadas a lógica de código.	*Com Justificativa*	19/58 (32,7%)	13/59 (22%)
	Sem Justificativa	15/58 (25,86%)	31/59 (52,54)%
	Total	34/58 (58,6%)	44/59 (74,57%)

4.1 Caracterização dos Pull Requests que induziram refatoramentos e dos Pull Requests que não induziram refatoramentos

4.1.1 Pull Requests que induziram refatoramentos

É possível notar que houve uma distribuição semelhante de cenários, no qual nenhum dos cenários obteve grande maioria, porém uma observação bastante interessante consta no fato de que *RIPRs*, em sua maioria(63,8% dos casos da amostra), possuíram algum tipo de revisão que indicou possíveis refatoramentos a serem realizados, independentemente do tipo, isso nos dá a idéia de que talvez, a ocorrência de *RIPRs* com comentários relacionados a refatoramentos seja relativamente comum.

Como exemplo de tal situação, a Figura 3 mostra um comentário de revisão em que há um pedido explícito de um refatoramento através da introdução da variável *expiredCheckpoint,* junto a uma justificativa para a mesma ser criada. Já na Figura 4, conseguimos visualizar um comentário de revisão com uma sugestão de mudança de nome do método *testTooSmallTimeSpan()* para *testTimeSpanBelowUpdateRate()*, sem a presença de justificativas para a mudança.

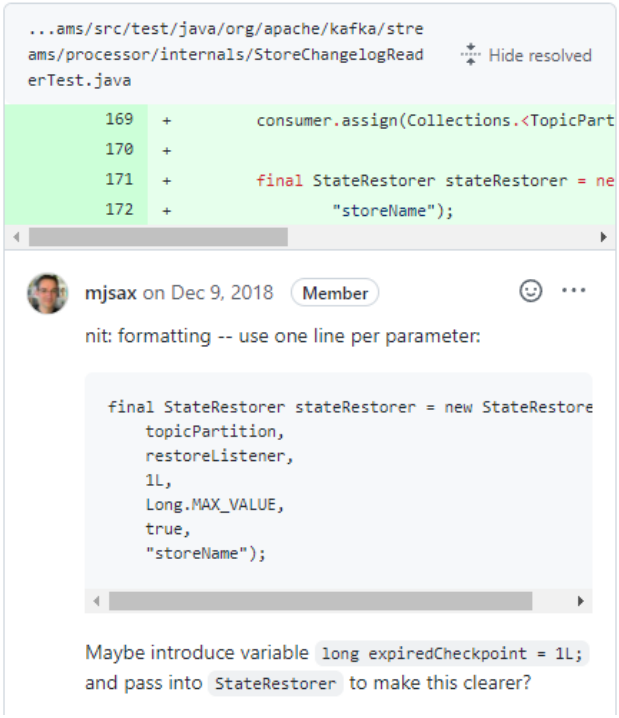

Figura 3: Exemplo de sugestão de refatoramento com justificativa em um Pull Request que induziu refatoramentos

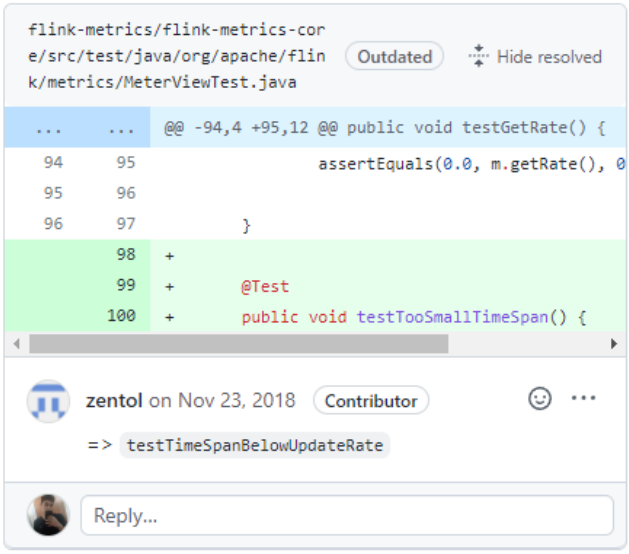

Figura 4: Exemplo de sugestão de refatoramento sem justificativa em um Pull Request que induziu refatoramentos

É possível notar também que segundo a amostra, os comentários que sugeriram refatoramentos relacionados à mudança de nome e os comentários que sugeriram outros tipos de refatoramento ocorreram em números bastante semelhantes(36,2% e 34,4% dos casos, respectivamente). Talvez, isso dê a ideia de que sugestões de refatoramentos que sugiram mudanças de nome, sejam o tipo mais comum de sugestão de refatoramento, afinal, elas ocorrem em número semelhante à sugestões de todos os outros tipos de refatoramento.

A Figura 5 exemplifica um comentário em que o revisor sugere uma mudança de nome da variável *addReferencedObject* para diferenciar ela da variável *addReferencedObjects* com a devida justificativa. Já a Figura 6 exemplifica um comentário que sugere a mudança do tipo de retorno da função de *Set* para *LinkedHashSet*, sem apresentar justificativas.

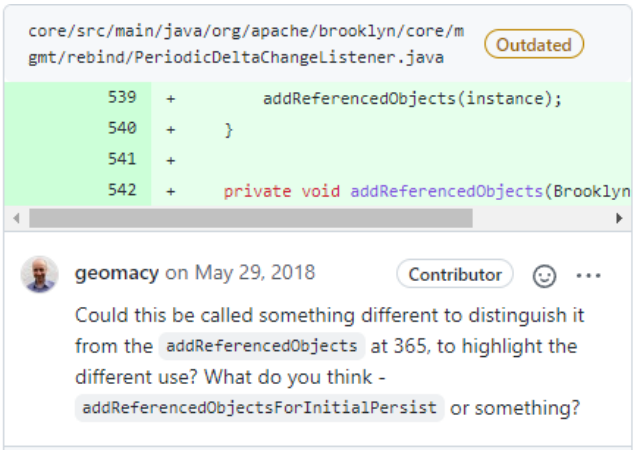

Figura 5: Exemplo de sugestão de refatoramento de mudança de nome em um Pull Request que induziu refatoramentos

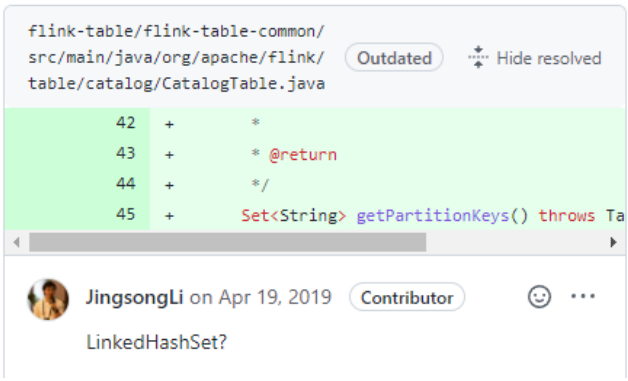

Figura 6: Exemplo de sugestão de refatoramento de mudança de tipo em um Pull Request que induziu refatoramentos

Os *RIPRs* da amostra também possuíram em sua maioria, comentários que sugeriram mudanças relacionadas a lógica de código(58,6% dos casos) como por exemplo, na Figura 7, onde é possível ver uma sugestão de possível problema na lógica do código, com um possível comportamento indevido no mesmo. Isso indica que, provavelmente, o processo de refatoramento de código geralmente pode ser acompanhado de mudanças de lógica em seu conjunto.

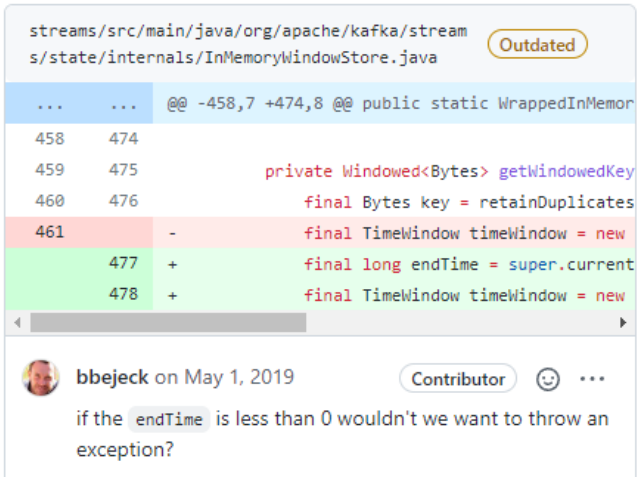

Figura 7: Exemplo de sugestão de mudança de lógica em um Pull Request que induziu refatoramentos.

De forma resumida, *RIPRs* apresentaram uma distribuição não muito dispersa dos cenários analisados, no qual todos os mesmos tiveram uma ocorrência na amostra que talvez indique a diversidade dos casos possíveis, podendo um *RIPR* possuir qualquer um dos cenários analisados.

Foi possível observar um padrão na amostra no qual a maioria(63,8% dos casos) dos *RIPRs* possuíram sugestões que tratavam de refatoramentos, isso pode significar que a ocorrência de refatoramentos em Pull Requests após sua abertura pode estar bastante relacionada com o fato de existirem sugestões nos comentários de revisão relacionadas a refatoramentos.

Também foi possível notar que provavelmente, o processo de sugestão de refatoramentos está geralmente acompanhado de sugestões de lógica, visto que a maioria(58,6% dos casos) dos *RIPRs* também possuíam sugestões relacionadas a mudanças na lógica de código.

4.1.2 Pull Requests que não induziram refatoramentos

Para os casos dos *non-RIPRs*, é possível notar um comportamento bastante comum e semelhante, em apenas 1,6% dos casos, houve sugestões relacionadas a refatoramentos. O único caso de sugestão de refatoramento tratou-se de uma sugestão de mudança de nome, não foi possível encontrar algum motivo evidente, visto que o autor apenas ignorou a sugestão. A Figura 8 ilustra o caso da sugestão ignorada, na qual o autor mencionou que o nome do método poderia soar estranho, e pediu a mudança do método para *canFuse*.

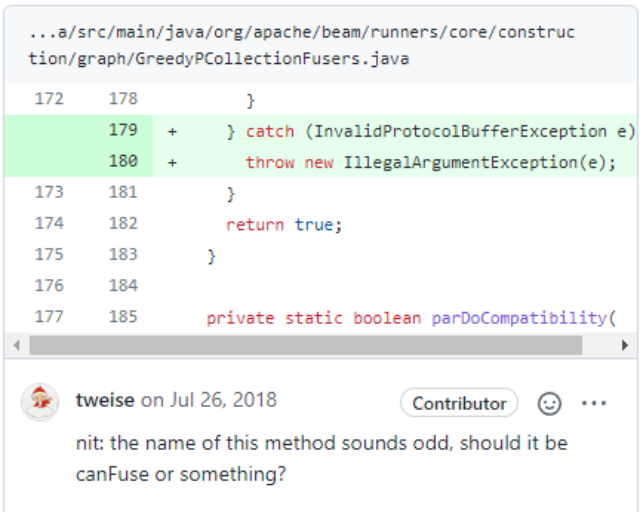

Figura 8: Sugestão de mudança de nome ignorado pelo autor do commit

Nenhum dos *non-RIPRs* possuíram algum comentário sugerindo refatoramentos que não envolvessem mudanças de nome.

Por outro lado, em 98,3% dos casos da amostra, ocorreram comentários de revisão que não tratavam de refatoramentos, com 74,5% dos casos possuindo algum tipo de sugestão relacionada a lógica de casos

Uma possível conclusão que pode ser feita é que a ocorrência de refatoramentos geralmente é acompanhada de sugestões, porém, não é possível inferir pela a amostra de que a ausência de comentários de revisão sugerindo refatoramentos garanta que não ocorram refatoramentos, visto que em um número considerável(36,2% dos casos da amostra) de *RIPR*s não houve comentários sugerindo refatoramentos, mas eles ocorreram mesmo assim.

4.2 Diferenças entre Pull Requests que induziram refatoramentos e Pull Requests que não induziram refatoramentos no nível de comentário de revisão

As diferenças entre os *RIPRs* e os *non-RIPRs* no nível de comentário de revisão são perceptíveis quando analisamos o conteúdo dos comentários.

Enquanto aproximadamente 36% dos *RIPRs* não tiveram mudanças relacionadas à refatoramentos, aproximadamente 98% dos *non-RIPRs* da amostra não tiveram comentários com sugestões relacionadas à refatoramentos. Porém, quando analisamos o quesito de comentários relacionados à mudanças de lógica de código, considerando toda a mostra, é possível notar números não tão discrepantes.

Enquanto os *RIPRs* apresentaram essa característica em 58,6% dos casos, os *non-RIPRs* apresentaram essa característica em 74,57% dos casos, com uma diferença de aproximadamente 15%. Porém, foi possível notar que os comentários de *RIPRs* relacionados à mudanças de código possuíram menos justificativas acompanhando as sugestões quando comparados com os *non-RIPRs*(25,8% e 52,54% dos casos, respectivamente).

Isso pode gerar a hipótese de que provavelmente sugestões de lógica de código em *non-RIPRs* são mais comuns de serem acompanhadas por justificativas, talvez pelo fato de que muitas vezes os revisores estão discutindo questões de refatoramento que podem acabar por serem complementadas pelas sugestões de mudanças de lógica. Na figura abaixo é possível ver um exemplo de comentário no qual o revisor sugeriu refatoramentos e também discutiu questões de lógica sobre comportamento do código.

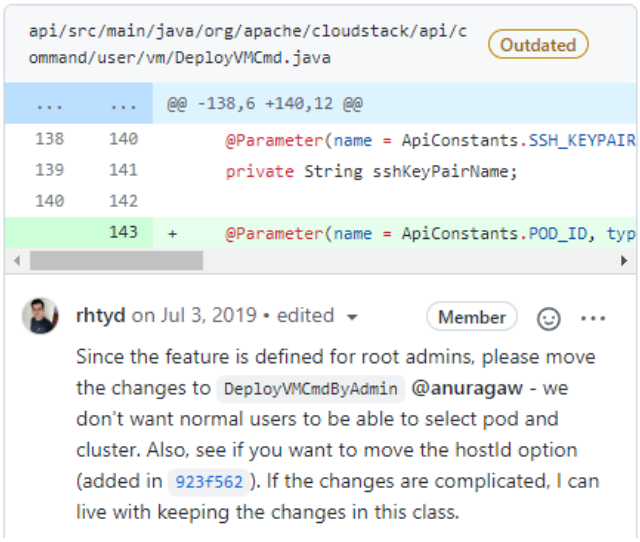

Figura 9: Comentário de revisão que justifica um possível refatoramento através da discussão da lógica e do comportamento do código, em um Pull Request que induziu refatoramentos

5. RECONHECIMENTOS

Essa pesquisa só foi possível, primeiramente, graças a ferramenta *RefactoringMiner[2]*, desenvolvido por Nikolaos Tsantalis, Ameya Ketkar e Danny Dig, pois a ferramenta automatizou a parte mais demorada do processo de análise, que era a fase inicial de coleta e classificação de dados. Agradecemos à Tiago Massoni(Universidade Federal de Campina Grande), Everton Alves(Universidade Federal de Campina Grande) e Flávia Coelho(Universidade Federal de Campina Grande), pela orientação e mentoria tanto no processo de pesquisa quanto no processo de escrita deste artigo, e também agradecemos novamente à Flávia Coelho, pois seu trabalho anterior[1], tornou possível e forneceu uma base para este trabalho.

6. CONSIDERAÇÕES FINAIS

Apesar da pesquisa ter obtidos resultados satisfatórios, talvez caso a pesquisa tivesse um prazo maior e uma equipe maior de pesquisadores disponível, as hipóteses pudessem ter sido testadas em uma amostra maior. As hipóteses obtidas nesse trabalho certamente nos ajudam com uma direção para futuras pesquisas, porém ainda são apenas hipóteses e para serem confirmadas, seria necessário uma amostra consideravelmente maior de Pull Requests analisados.

7. REFERÊNCIAS

[1] Flávia Coelho, Nikolaos Tsantalis, Tiago Massoni and Everton L. G. Alves. Characterizing Refactoring-Inducing Pull Requests.

[2] About Github. https://github.com/about. Acessado em 09/10/2021.

[3] About Git. https://git-scm.com/about. Acessado em 09/10/2021.

[4] The Apache® Software Foundation projects statistics. https://projects. apache.org/statistics.html, Novembro 2020. Acessado em: Dezembro 2020.

[5] Nikolaos Tsantalis, Ameya Ketkar, and Danny Dig. RefactoringMiner 2.0. IEEE Transactions on Software Engineering, 2020.

[6] Nikolaos Tsantalis, Matin Mansouri, Laleh M. Eshkevari, Davood Mazinanian, and Danny Dig. Accurate and efficient refactoring detection in commit history. In Proceedings of the 40th International Conference on Software Engineering, ICSE'18, pages 483–494, Gothenburg, Sweden, 2018. ACM.

[7] Github Pull Request Flow. https://www.researchgate.net/figure/GitHub-Pull-request-flow_fig1_326295010. Acessado em 10/10/2021.

[8] What is a Pull Request in Git?. https://www.gitkraken.com/learn/git/tutorials/what-is-a-pull-request-in-git#:~:text=A%20pull%20request%20is%20an,remote%20hosting%20service%2C%20like%20GitHub. Acessado em 08/10/2021.