Análise de focos de retrabalho em repositórios de software: um estudo em projetos do GitHub

Gustavo Graf de Sousa
Universidade Federal de Viçosa - Campus Florestal
Florestal, Minas Gerais
graf.sousa@gmail.com

Gláucia Braga e Silva
Universidade Federal de Viçosa - Campus Florestal
Florestal, Minas Gerais
bragaesilva@gmail.com

ABSTRACT

O presente trabalho realiza a análise conjunta de dados em ambientes de rastreamento de questões (*issue tracking*) e controle de versão. O objetivo é identificar possíveis focos de retrabalho em projetos de software, em duas categorias de *issues*: as que possuem código associado e aquelas que, embora tenham envolvido muitas discussões, não convergiram para código. Os dados foram extraídos de projetos armazenados na plataforma *GitHub*. Para isso, foram aplicadas análises estatísticas tais como: correlação; regressão múltipla e teste de hipótese. As análises foram realizadas sobre os dados relacionados aos seguintes parâmetros: *número de commits*, *número de comentários*, *número de desenvolvedores* e *número de reaberturas*. Esses parâmetros foram analisados de forma conjunta, para se verificar quais eram as influências existentes entre eles. Os resultados obtidos mostraram que os parâmetros selecionados podem ter forte relação com focos de retrabalho nas duas categorias de *issues* avaliadas.

CCS CONCEPTS

• **Software and its engineering → Software configuration management and version control systems**; **Software evolution**; **Software version control**; *Software libraries and repositories*; Maintaining software;

KEYWORDS

Issue Tracking, Version Control, Supplementary Bug Fixes, Statistical Analyzes, Reopened Bugs

ACM Reference Format:
Gustavo Graf de Sousa and Gláucia Braga e Silva. 2019. Análise de focos de retrabalho em repositórios de software: um estudo em projetos do GitHub. In *Proceedings of CBSoft 2019 (SBES2019).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUÇÃO

No contexto de um processo de software, retrabalho é todo esforço utilizado para refazer uma tarefa que foi realizada de forma incorreta, podendo ser causado pela falha na comunicação, pelo não entendimento da tarefa a ser feita, por especificações incompletas e/ou incorretas ou por mudanças ao longo do processo. Retrabalho tem um grande impacto no custo e no prazo, uma vez que consome recursos necessários a outras tarefas do desenvolvimento. Estima-se que 25% a 40% do esforço no desenvolvimento de um software é gasto em retrabalho e 70% a 80% do custo do retrabalho é devido a erros nas especificações dos requisitos [9]. Assim, do ponto de vista gerencial, torna-se fundamental identificar e quantificar o retrabalho em um projeto de software, rastreando onde ele ocorre, artefatos mais afetados, desenvolvedores envolvidos e tempo gasto [2, 8].

Para realizar as análises sobre retrabalho, repositórios de Gerência de Configuração de Software (GCS) constituem importantes fontes de dados já que armazenam o histórico do trabalho da equipe ao longo do processo. A GCS tem como principais objetivos controlar as mudanças que ocorrem nos itens de configuração e as versões correspondentes. O processo de GCS abrange todo o ciclo da mudança, desde a identificação de uma *issue*, passando pela avaliação da mesma até a execução da mudança e a geração de uma nova versão aprovada do item de configuração. Ao analisar os dados de GCS, pode-se avaliar quanto esforço foi gasto para corrigir essa mudança, quantos recursos foram utilizados, se a mudança é recorrente, quantas versões foram geradas, dentre outras questões que podem indicar focos de retrabalho neste contexto [1, 11].

Este trabalho tem como objetivo a análise conjunta de dados em ambientes de rastreamento de questões (*issue tracking*) e controle de versão para identificação de focos de retrabalho, considerando esforços relacionados a *issues* com produção de código correspondente ou não. Para isso, serão aplicadas análises estatísticas sobre dados relacionados aos seguintes parâmetros: número de vezes que a *issue* foi reaberta, número de edições recorrentes de um mesmo artefato em uma *issue*, número de envolvidos em uma *issue* e número de mensagens trocadas em uma *issue*. Como prova de conceito da análise proposta, serão avaliados dados de projetos armazenados no *GitHub*, que possuam informações de *issue tracking* e controle de versão relacionadas, ou seja, só serão analisados dados em que as modificações nos artefatos tenham sido analisadas via *issue*.

A análise desenvolvida poderá ajudar gerentes de projetos a identificar possíveis focos de retrabalho ao longo do desenvolvimento de um projeto, verificando se existe alguma relação entre eles e os parâmetros de análise selecionados. Os resultados da análise podem revelar a influência de um ou mais parâmetros na ocorrência de retrabalho, como por exemplo, a relação de reaberturas em uma *issue* e o número de *commits* associados ou discussões recorrentes em uma *issue*, envolvendo vários desenvolvedores, mas que não convergem para código (sem *commit* associado) e que possuam ou não reabertura. Diante disso, os gerentes podem definir estratégias

SBES2019, Setembro 2019, Salvador, BA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

para melhorar a qualidade das especificações, com o intuito de reduzir a quantidade de *commits* relacionados a uma *issue* e o número de reaberturas e buscar meios de melhorar a qualidade das mensagens trocadas, com o intuito de reduzir falhas na comunicação, reduzindo o retrabalho e consequentemente, os custos do projeto.

O presente artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação acerca de retrabalho no contexto de desenvolvimento de software. Na Seção 3, são apresentadas as análises estatísticas aplicáveis. Os trabalhos relacionados são discutidos na Seção 4. As medições para identificar focos de retrabalho em repositórios de software são discutidas na seção 5 e na seção 6, apresenta-se a conclusão do trabalho.

2 RETRABALHO NO CONTEXTO DE DESENVOLVIMENTO DE SOFTWARE

Segundo o padrão Systems and software engineering–Vocabulary [6], "Retrabalho é a ação tomada para levar à conformidade com os requisitos ou especificações um componente não conforme ou defeituoso (tradução nossa)".

Em um processo iterativo e incremental de desenvolvimento de software, retrabalho é inevitável. Isto acontece pois a cada nova iteração, pode ser necessário corrigir e/ou refinar uma funcionalidade. Além disto, bugs são encontrados na fase de testes e precisam ser reparados, os requisitos são alterados e novas funcionalidades são incluídas [5, 10]. No entanto, a quantidade de retrabalho é um indicativo de como encaminha-se o progresso do software visto que uma quantia exorbitante pode demonstrar a falta de clareza nos requisitos, e uma pequena quantidade pode indicar pouca revisão e testes. [5]

Para ser capaz de avaliar o retrabalho no desenvolvimento de software, é necessário definir métricas para estimá-lo. Usualmente, a quantidade de código fonte modificado, a alteração nos requisitos e na arquitetura, mudanças recorrentes para uma *issue*, entre outras, são utilizadas para a estimativa total do retrabalho [1, 7, 11].

Fairley e Willshire [5] definem dois tipos de retrabalho: o evolutivo (*evolutionary*) e o evitável (*avoidable*). O segundo pode ser subdividido em retrospectivo (*retrospective*) e corretivo (*corrective*). Essas categorias de retrabalho são categorizadas em bom (*good*), ruim (*bad*) e terrível (*ugly*), de acordo com a quantia de retrabalho (Tabela 1).

Neste trabalho serão avaliados e analisados somente focos de retrabalho evitáveis, como por exemplo, os *Supplementary Bug Fixes*, porque são os que mais causam impacto no desenvolvimento de software uma vez que normalmente são correções de defeitos ou tarefas recorrentes, em virtude de especificações incorretas e/ou incompletas.

2.1 Retrabalho e o Ciclo da Mudança

A etapa de gerenciamento de mudanças tem como finalidade realizar o acompanhamento das mudanças solicitadas pelos *stakeholders*, determinar quando elas serão incluídas e definir os custos e impactos das mesmas. Ferramentas automatizadas são utilizadas para ajudar no gerenciamento das mudanças requisitadas para garantir que elas sejam aplicadas de forma controlada no sistema [10].

Conforme mostra a Figura 1, o ciclo de vida da mudança é iniciado quando uma *issue* é identificada, com o intuito de registrar alguma demanda de modificação em um artefato de software. É realizado um registro (registro da *issue*) formalizando essa solicitação, contendo informações de suporte relevantes como detalhes da *issue* identificada, origem e ICs (itens de configuração) envolvidos. Na etapa de avaliação da *issue*, primeiramente é realizada uma análise da mudança solicitada, podendo esta ser aprovada ou não. Caso seja reprovada, o solicitante é notificado e encerra-se o processo. Sob a condição de aprovação, denomina-se o responsável para efetuar as mudanças requisitadas [3]. Ao iniciar a etapa de execução da mudança, é realizado um *checkout* dos ICs relacionados à *issue* reportada para que as alterações sejam aplicadas nos mesmos. No início da etapa de execução da mudança, são realizadas cópias (*checkout*) das versões atuais dos ICs envolvidos referentes à *issue* aprovada. As modificações solicitadas são efetuadas e futuramente são inspecionadas com o intuito de verificar se as mudanças foram corretamente implementadas. Em caso de reprovação, a alteração é revisada e auditada novamente. Caso ela seja aprovada, a nova versão dos ICs são devolvidas em uma operação de *checkin* [3].

Este trabalho busca nas três fases do processo da mudança por focos de retrabalho para duas categorias de *issues*: as que não contém mudanças aplicadas (sem código correspondente) e as que contém mudanças recorrentes. Para a categoria de *issues* que não possuem código correspondente, serão analisados dados das fases de Identificação e Avaliação das Issues, porque embora ela não tenha gerado código, houve alocação de recursos humanos que demandaram esforço acerca das discussões relacionadas, mas ou a *issue* não foi aprovada ou foi abandonada/esquecida. Já para a categoria de *issues* que geraram código, ou seja, a mudança foi aprovada, serão analisados os dados da fase de Execução da Mudança, avaliando ainda todos os *commits* relacionados àquela *issue*. Nesse último caso, a literatura já aponta que existe retrabalho quando se tem 2 ou mais *commits* associadas a uma *issue*. Essas mudanças recorrentes são chamadas de *Supplementary Bug Fixes* (SBF) [1]. SBF enquadram-se na definição de retrabalho evitável retrospectivo/corretivo, proposta por [5]. Da mesma forma, apesar de ser uma avaliação controversa na literatura, a presença de reaberturas, seja nas fases de Identificação e Avaliação das *Issues* ou na fase de Execução da Mudança, também pode indicar retrabalho. Souza [4], ao investigar retrabalho utilizando como parâmetro mudanças inapropriadas, conclui que há motivos para não considerar reabertura como um parâmetro para se avaliar retrabalho, pois muitas reaberturas acontecem porque o desenvolvedor reabre a *issue* por engano. An, Khomh e Adams [1] apresentam que 21.6% a 33.8% dos SBF são reabertos. Por outro lado, o trabalho também encontrou que 57.5% das *issues* reabertas não fazem parte do conjunto de SBF, pois possuem somente um *commit* associado, podendo ter uma forte associação com reaberturas inválidas.

3 ANÁLISES ESTATÍSTICAS APLICÁVEIS

Neste trabalho, os dados armazenados em repositórios de GCS serão analisados com base em técnicas estatísticas como correlação, regressão múltipla e teste de hipótese.

A correlação é um método estatístico no qual tenta-se entender como as variáveis se comportam em um cenário, ou seja, através dela é possível verificar se dois fenômenos estão relacionados. Ela pode ser positiva, quando a variação dos parâmetros avaliados está

Tabela 1: Uma taxonomia de retrabalho iterativo

Tipos de Retrabalho	Características	Bom, ruim ou terrível ?
Evolutivo	Trabalho realizado em uma versão anterior do software no intuito de aprimorar e agregar valor	Bom - Se adiciona valor sem violar o custo ou atrasar o cronograma. Ruim - Se violar o custo ou atrasar o cronograma Terrível - Se possui muito "gold plating".
Evitável - Retrospectivo	Trabalho realizado em uma versão anterior do software na qual os desenvolvedores deveriam ter feito anteriormente	Bom - Pequenas quantidades são inevitáveis; antes agora do que mais tarde. Ruim - Se ocorre rotineiramente. Terrível - Se é excessivo, indica que é necessário revisar o processo.
Evitável - Corretivo	Trabalho realizado para corrigir defeitos em versões antigas e na atual de um software.	Bom - Se o retrabalho total está dentro dos limites. Ruim - Se resulta em padrões de causa-efeito. Terrível - Se resulta em um processo de desenvolvimento descontrolado.

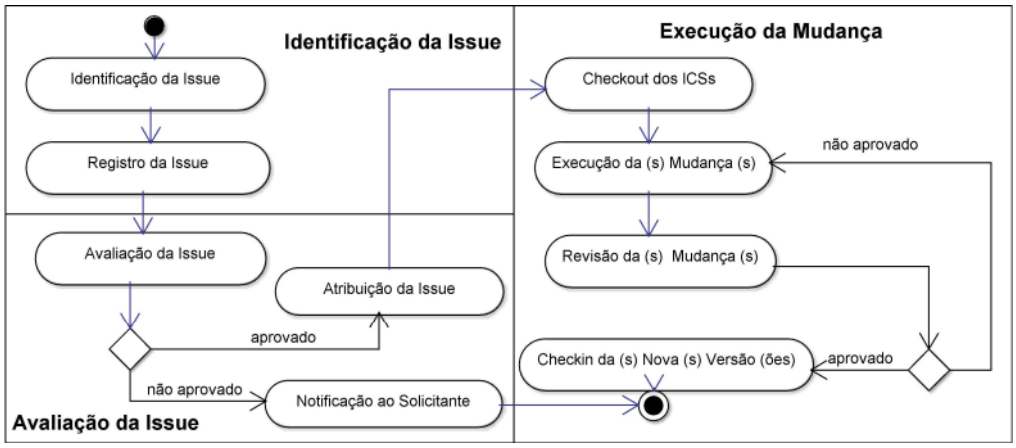

Figura 1: Fases do Processo da Mudança (Adaptado de Braga e Silva[3])

no mesmo sentido ou negativa, quando os variações dos parâmetros está em sentidos opostos. O coeficiente de correlação varia entre -1 a 1, sendo que, quanto mais próximo de 1 ou -1, mais forte é a associação, e quanto mais próximo de 0, menor é a associação dos dados [12]. De acordo com Charles [12], a ferramenta estatística de regressão linear "permite quantificar a relação entre uma variável específica e um resultado de interesse enquanto outros fatores são controlados". Ao se adicionar várias variáveis na equação linear, a análise mostra como as variáveis explicativas impactam na variável dependente [12]. Um coeficiente de muita importância ao se analisar uma regressão é o coeficiente de determinação, o R^2. Ele indica quanto o modelo de regressão pode explicar os dados sob análise, podendo variar entre 0 a 1, sendo que quanto mais próximo de 1, mais ele explica a amostra. Por exemplo, um coeficiente de 0.95, explicaria 95% da amostra, enquanto um R^2 de 0.1 explicaria somente 10%.

Por fim, com o teste de hipóteses é possível aceitar ou rejeitar explicações com base em estatísticas [12]. Para a rejeição ou não da hipótese nula com resultados "estatisticamente significativos" , determina-se um nível de significância de 5%, que é um nível de significância razoável [12]. Para rejeição ou não da hipótese nula foi escolhido como parâmetros o intervalo de significância e o valor P. Através do cálculo e comparação dos intervalos de confiança tem-se indícios dos comportamentos analisados. Caso haja sobreposição entre os intervalos, não se pode inferir nada sobre os dados. O cálculo do valor P vêm para reforçar o intervalo de confiança. Devido ao nível de significância ser de 5%, caso o valor P encontrado seja menor que 0.05, pode-se rejeitar a hipótese nula. Um P valor muito pequeno indica um evento muito incomum.

4 TRABALHOS RELACIONADOS

Nesta seção são apresentados diversos trabalhos que servirão de inspiração e referência para o desenvolvimento desta pesquisa.

Em uma abordagem apresentada por Rúbio e Gulo [11], é realizada uma análise de *commits* em projetos *Open Source* armazenados no *GitHub*, utilizando de metodologias de mineração de dados para estabelecer uma classificação do retrabalho de desenvolvedores. Foram utilizados três algoritmos diferentes, *Decision Trees*, *Naive Bayes* e *K-NM* e realizada uma comparação entre eles na qual o algoritmo *K-NM* apresenta os melhores resultados. O objetivo principal é identificar se um *commit* é foco de retrabalho ou não de acordo com as informações das atividades dos desenvolvedores. Para a verificação do modelo proposto, foram analisados 3311 *commits*. Com uma taxa de 70%, o modelo conseguiu predizer se um *commit* é uma fonte de retrabalho ou não. Por outro lado, a análise proposta neste trabalho se baseia em técnicas estatísticas com o intuito de identificar outros focos de retrabalho em repositórios do *GitHub*, utilizando de outros parâmetros, como por exemplo, número de *commits* associados a uma *issue* e número de reaberturas.

An, Khomh e Adams [1] realizaram uma pesquisa com o intuito de compreender a relação entre *Supplementary Bug Fixes* (SBF) e *issues* que tiveram ao menos uma reabertura. SBF's são *issues* que possuem várias modificações, ou seja, mais de um *commit* associado. Foi encontrado que 21.6% a 33.8% dos SBF foram reabertos ao menos uma vez, comprovando que a reabertura é um motivo importante na origem de SBF. O presente trabalho aprofunda na relação SBF x Reabertura, adicionando outros parâmetros como o número de desenvolvedores envolvidos e número de comentários em uma *issue*, identificando assim, outros focos de retrabalho. Além disso, aqui serão avaliados dados de projetos armazenados no *GitHub* que possuam informações de *issue tracking* e controle de versão relacionadas entre si.

Souza [4] realizou um estudo para compreender a rejeição de mudanças e como elas resultam em retrabalho nos projetos. Para alcançar os objetivos, foi realizada uma análise integrada entre *issues* e *commits* utilizando a técnica de detecção de mudanças rejeitadas. Para identificá-las foram definidas métricas a partir do número de *issues* com uma revisão negativa, número de *issues* revertidos e número de *issues* reabertos. Apesar da presente pesquisa também realizar a análise conjunta entre *issues* e *commits* e de também avaliar reaberturas como possíveis focos de retrabalho, a mesma não faz uma análise específica sobre mudanças aprovadas ou rejeitadas.

5 MEDIÇÕES PARA IDENTIFICAR FOCOS DE RETRABALHO EM REPOSITÓRIOS DE SOFTWARE

Esta seção apresenta as medições realizadas sobre dados de GCS e as análises estatísticas aplicadas.

Com a finalidade de identificar e avaliar possíveis focos de retrabalho, para cada *issue*, foram mapeados os seguintes parâmetros: número de *commits* associados, número de comentários, número de envolvidos e número de reaberturas.

5.1 Obtenção dos dados

Para coletar os dados em repositórios armazenados no *GitHub*, foi utilizada a *API* do *GitHub* para a linguagem de alto nível Java [1]. Para realizar o armazenamento dos dados foram criados arquivos com a extensão *.xls* (*Excel*) a partir da biblioteca *POI* disponibilizada pelo *Apache* [2].

Para a realização das análises estatísticas foi utilizado a linguagem de programação *Python*, versão 6.2.1, em específico a sua biblioteca *Pandas*. A ferramenta utilizada como *IDE* para a fazer estas análises foi o *Jupyter Notebook*, versão 4.4.0[3].

Através da API do *GitHub* foi possível realizar a coleta das *issues* e os parâmetros avaliados. Para contabilizar o número de *commits associados* a uma *issue* foi verificado se na descrição do *commit* havia o número da *issue* que ele era destinado. Em caso de positivo, era contabilizado um *commit associado* a *issue*. A partir de métodos específicos da *API* do *GitHub* é possível coletar os comentários de uma *issue* e com isto determinar o *número de comentários* e o *número de envolvidos*. O *número de envolvidos* em uma *issue* foi contabilizado através dos diferentes autores dos comentários daquela *issue*. Para o cálculo do *número de reaberturas* foi analisado os eventos da *issue*. Quando um evento do tipo *"reopened"* era encontrado, era contabilizado uma reabertura para aquela *issue*. Os dados coletados foram armazenados em um arquivo com a extensão *.xls* que foi criado através da biblioteca *POI*. Este arquivo foi lido na ferramenta *Jupyter Notebook* utilizando a linguagem *Python*, em específico a biblioteca *Pandas*.

5.2 Análises estatísticas realizadas

Foram extraídos um total de 111575 *issues* de 17 projetos armazenados no *GitHub*. Para tentar compreender melhor o comportamento das *issues* obtidas, foram realizadas estatísticas descritivas que resumem a tendência central, a dispersão e a forma da distribuição do conjunto de dados. Os resultados obtidos foram sintetizados na Figura 2. Nesta figura é mostrado a média (*mean*), o desvio padrão (*std*), o valor mínimo e máximo (*min* e *max*) encontrados, e os percentis (50% e 95%) para os parâmetros avaliados.

Para se ter certeza do grau de relacionamento entre os parâmetros, foram calculadas as correlações entre eles os dados conforme ilustra a Figura 3. Analisando a tabela pode-se perceber que a correlação entre a maioria dos dados é baixa, sendo a maior correlação de 79.3% entre a quantidade de comentários e usuários envolvidos. Outro dado interessante é a correlação negativa, baixa (16.5%), entre quantidade de usuários envolvidos com quantidade de *commits* associados.

Devido à baixa correlação entre os dados, espera-se que o modelo de regressão linear explique pouco os dados, ou seja, o r^2 será baixo. Ao fixar a quantidade de *commits* associados e o número total de comentários, o r^2 possui um valor baixo (0.062) e (0.622). Isto quer dizer que a regressão explica somente 6.2% dos dados em relação aos *commits* associados e 62.2% em relação aos comentários. Através dos coeficientes (coefs), pode-se perceber que, por exemplo, caso todos os parâmetros sejam zero, o número de *commits* associados será de 0.56 e o total de comentários será de -0.64 (linha *Intercept*). Além disso, pode-se observar que devido a ser o maior coeficiente, o número de reaberturas é o que mais causa impacto em ambos os casos, ou seja, um número alto de reaberturas tende a aumentar o número de *commits* e comentários associados a uma *issue* (Figura 4).

[1] Download API GitHub

[2] Download POI

[3] Download Python e Jupyter

	QtdDeCommitsAssociados	TotalComentarios	QtdDeUsuariosEnvolvidos	QtdDeReaberturas
count	111575.000000	111575.000000	111575.000000	111575.000000
mean	0.429881	3.344450	1.877006	0.000798
std	0.603786	5.133018	2.004300	0.034256
min	0.000000	0.000000	0.000000	0.000000
50%	0.000000	2.000000	2.000000	0.000000
95%	1.000000	12.000000	5.000000	0.000000
max	44.000000	198.000000	71.000000	5.000000

Figura 2: Descrição dos dados totais

	QtdDeCommitsAssociados	TotalComentarios	QtdDeUsuariosEnvolvidos	QtdDeReaberturas
QtdDeCommitsAssociados	1.000000	-0.101850	-0.165866	0.084820
TotalComentarios	-0.101850	1.000000	0.793346	0.044414
QtdDeUsuariosEnvolvidos	-0.165866	0.793346	1.000000	0.028581
QtdDeReaberturas	0.084820	0.044414	0.028581	1.000000

Figura 3: Correlações entre os Dados

```
Dep. Variable:     QtdDeCommitsAssociados
Model:                               OLS
R-squared:                         0.062
Adj. R-squared:                    0.062

                                    coef
-------------------------------------
Intercept                         0.5655
TotalComentarios                  0.0097
QtdDeReaberturas                  1.5412
QtdDeUsuariosEnvolvidos          -0.0679
```

```
Dep. Variable:       TotalComentarios
Model:                            OLS
R-squared:                      0.622
Adj. R-squared:                 0.622

                                    coef
-------------------------------------
Intercept                        -0.6430
QtdDeCommitsAssociados            0.2854
QtdDeReaberturas                  3.1952
QtdDeUsuariosEnvolvidos           2.0474
```

Figura 4: Regressão Linear - Fixando Commits e Comentários

Como o método estatístico de regressão não mostrou correlação e não foi capaz de explicar a maioria dos dados, e considerando-se que retrabalho é um comportamento não padrão (exceção), valores acima dos encontrados na Figura 2, ao nível de 95%, foram considerados como *outliers* e foram analisados separadamente. Os dados foram então separados nos seguintes grupos:

- G1.1: Dados Gerais - Sem Outliers de Commits;
- G1.2: Dados Gerais - Sem Outliers de Comentários;
- G1.3: Dados Gerais - Sem Outliers de Usuários Envolvidos;
- G1.4: Dados Gerais - Sem Outliers de Reaberturas;
- G2: Outliers de Commits (SBF);
- G3: Outliers de Comentários;
- G4: Outliers de Usuários Envolvidos;
- G5: Outliers de Reaberturas.

Para cada grupo de *outliers*, foram analisados cada parâmetro mapeado separadamente e comparado com o respectivo parâmetro de G1.X, com o intuito de verificar se a amostra possui comportamento similar ou não aos do grupo de dados gerais sem o respectivo grupo de *outlier*. Por exemplo, para G2, foram realizadas análises sobre a quantidade total de comentários, número de envolvidos e quantidade de reaberturas e os resultados obtidos foram comparados com os resultados de G1.1. Essa análise foi repetida para os demais grupos, alterando-se os parâmetros analisados e o grupo G1.X.

Por fim, para conduzir as análises foram formuladas as seguintes hipóteses:

- **Hipótese Nula:** As amostras analisadas (*outliers*) não são focos de retrabalho e seu comportamento não se difere dos do comportamento dos demais dados;
- **Hipótese Alternativa:** As amostras analisadas (*outliers*) são focos de retrabalho e seu comportamento se difere do comportamento dos demais dados.

5.2.1 Criação e análise de G1.1 Dados Gerais - Sem Outliers de Commits. Após a remoção dos *outliers de commits*, *issues* com mais de um *commit associado* (G2), o número total de *issues* foi para 110215. Na Figura 5 novamente são apresentadas estatísticas descritivas que resumem a tendência central, a dispersão e a forma da distribuição do conjunto de dados.

5.2.2 Analisando G2 - Outlier de Commits (SBF). O grupo G2, *outliers* de *commits*, é constituído de *issues* que possuem mais de um *commit* associado. Ou seja, todo SBF - *Supplementary Bug Fixes* pertence ao grupo G2. Ele contém 1360 *issues*, representando um

	QtdDeCommitsAssociados	TotalComentarios	QtdDeUsuariosEnvolvidos	QtdDeReaberturas
count	110215.000000	110215.000000	110215.000000	110215.0
mean	0.400798	3.321835	1.872268	0.0
std	0.490062	5.056240	1.983125	0.0
min	0.000000	0.000000	0.000000	0.0
50%	0.000000	2.000000	2.000000	0.0
95%	1.000000	11.000000	5.000000	0.0
max	1.000000	198.000000	68.000000	0.0

Figura 5: Descrição dos dados - pós limpeza de outliers de commits

total de 1.21% do conjunto total. Apesar de ser um conjunto pequeno, foram realizadas análises para os parâmetros selecionados para tentar compreender o comportamento desses dados.

A Tabela 2 contêm os resultados encontrados das análises em G2 e a comparação com o grupo *G1.1*. Pode-se inferir através dos resultados obtidos que uma das possíveis causas de SBF seja o resultado da falta de qualidade das mensagens trocadas ou por deficiência nas especificações, pois mesmo havendo muita discussão em torno da *issue* foi necessário diversos *commits* suplementares. É interessante notar que mesmo havendo vários *commits associados* G2 quase não possui reaberturas. Uma das possíveis causas para o baixo número de reaberturas seja a grande quantidade de discussão em volta da *issue* com vários usuários envolvidos.

An, Khomh e Adams [1] mostraram que 21.6% a 33.8% dos SBF possuem reabertura. Porém ao analisar as reaberturas do conjunto G2 foi encontrado que 94.49% dos SBF não possuem nenhuma reabertura, valor que difere-se do encontrado por [1]. A diferença dos percentuais encontrados pode ter relação com a quantidade e tamanho dos projetos analisados ou com as características do projeto em si, incluindo número e perfil dos contribuidores/colaboradores e origem e natureza do projeto.

5.2.3 Criação e análise de G1.2 Dados Gerais - Sem Outliers de Comentários. Após a remoção dos outliers de comentários,*issues* com mais de doze comentários (G3), o número total de *issues* foi para 106837. Na Figura 6 novamente são apresentadas estatísticas descritivas que resumem a tendência central, a dispersão e a forma da distribuição do conjunto de dados.

5.2.4 Analisando G3 - Outliers de Comentários. O grupo G3 é formado por *issues* que possuem mais de 12 comentários. Esse grupo é constituído por 4738 *issues*, representando 4.24% do conjunto total de dados.

Na Tabela 3 pode-se visualizar os resultados encontrados das análises em G3 e a compará-los com os resultado obtidos com o grupo G1.2. Por meio desses resultados, pode-se concluir que há um desperdício em termos de recursos alocados, uma vez que tem-se muitos desenvolvedores envolvidos e muita discussão em torno da *issue*, porém a discussão não convergiu para código. Essa ausência de código explicaria o baixo número de reaberturas das *issues*, já que não existem (*commits*) associados.

5.2.5 Criação e análise de G1.3 Dados Gerais - Sem Outliers de Usuários Envolvidos. Após a remoção dos outliers de usuários envolvidos,*issues* com mais de cinco usuários envolvidos (G4), o número total de *issues* foi para 107583. Na Figura 7 novamente são apresentadas estatísticas descritivas que resumem a tendência central, a dispersão e a forma da distribuição do conjunto de dados.

5.2.6 Analisando G4 - Outliers de Usuários Envolvidos. O grupo G4 é formado por um total de 3992 *issues*, representando 3.57% dos dados gerais. Fazem parte deste grupo *issues* que possuem mais de 5 usuários envolvidos.

Na Tabela 4 pode-se visualizar os resultados obtidos através das análises e compará-los com os resultados de G1.3. Os resultados obtidos são parecidos com os valores do grupo G3, o que pode ser explicado pela alta correlação entre os parâmetros *número de desenvolvedores* e *número de comentários* (Figura 3).

5.2.7 Criação e análise de G1.4 Dados Gerais - Sem Outliers de Reaberturas. Após a remoção dos outliers de reaberturas,*issues* com ao menos uma reabertura (G5), o número total de *issues* foi para 111500. Na Figura 8 novamente são apresentadas estatísticas descritivas que resumem a tendência central, a dispersão e a forma da distribuição do conjunto de dados.

5.2.8 Analisando G5 - Outliers de Reaberturas. O conjunto de *outliers* de reaberturas é constituído de *issues* que possuem ao menos uma reabertura. Ele contém um total de 75 *issues*, representando 0.06% do conjunto total.

Apesar de ser um conjunto de dados muito pequeno, foram realizadas análises para tentar compreender o comportamento destes dados. Foi encontrado que todos os *issues* (100%) que possuem ao menos uma reabertura são *Supplementary Bug Fix* (G2). Dessa forma, apesar de ser um conjunto pequeno de dados, não se pode ignorar que reaberturas possam ter relação com retrabalho, como sugere Souza [4], uma vez SBF são reconhecidos na literatura como retrabalho[1].

Na Tabela 5 pode-se observar os resultados obtidos através das análises e compará-los com os resultos de G1.4.

5.2.9 Intervalo de Confiança e Valor P. Para comprovar que as amostras se diferem da população, ou seja, que o comportamento dos grupos G2 a G5 divergem em relação aos grupos G1.X, foi realizado o teste de hipótese. Para a aceitação ou rejeição da hipótese nula, foi calculado o intervalo de confiança e o valor p. As Tabelas

Tabela 2: G2 x G1.1

DataFrame/Parâmetros	Comentários (95% dos dados)	Usuários Envolvidos (95% dos dados)	Reaberturas
G2 - Outliers de Commits (SBF)	19	6	94.49% não possuem
G1.1 - Dados Gerais - Sem Outliers de Commits	11	5	100% não possuem

	QtdDeCommitsAssociados	TotalComentarios	QtdDeUsuariosEnvolvidos	QtdDeReaberturas
count	106837.000000	106837.000000	106837.000000	106837.000000
mean	0.434194	2.566648	1.660895	0.000534
std	0.600325	2.680220	1.408342	0.025037
min	0.000000	0.000000	0.000000	0.000000
50%	0.000000	2.000000	1.000000	0.000000
95%	1.000000	8.000000	4.000000	0.000000
max	44.000000	12.000000	12.000000	2.000000

Figura 6: Descrição dos dados - pós limpeza de outliers de comentários

Tabela 3: G3 x G1.2

DataFrame/Parâmetros	Commits	Usuários Envolvidos (95% dos dados)	Reaberturas
G3 - Outliers de Comentários	72.01% não possuem	15	99.51% não possuem
G1.2 - Dados Gerais - Sem Outliers de Comentários	95% possuem até 1	4	99.95% não possuem

	QtdDeCommitsAssociados	TotalComentarios	QtdDeUsuariosEnvolvidos	QtdDeReaberturas
count	107583.000000	107583.000000	107583.000000	107583.000000
mean	0.438303	2.815845	1.629477	0.000669
std	0.601120	3.653648	1.296781	0.032259
min	0.000000	0.000000	0.000000	0.000000
50%	0.000000	2.000000	1.000000	0.000000
95%	1.000000	9.000000	4.000000	0.000000
max	44.000000	124.000000	5.000000	5.000000

Figura 7: Descrição dos dados - pós limpeza de outliers de usuários envolvidos

Tabela 4: G4 x G1.3

DataFrame/Parâmetros	Commits	Comentários (95% dos dados)	Reaberturas
G4 - Outliers de Usuários Envolvidos	84.34% não possuem	40	99.60% não possuem
G1.3 - Dados Gerais - Sem Outliers de Usuários Envolvidos	95% possui até 1	9	99.94% não possuem

	QtdDeCommitsAssociados	TotalComentarios	QtdDeUsuariosEnvolvidos	QtdDeReaberturas
count	111500.000000	111500.000000	111500.000000	111500.0
mean	0.428377	3.339013	1.875318	0.0
std	0.599457	5.123497	2.001463	0.0
min	0.000000	0.000000	0.000000	0.0
50%	0.000000	2.000000	2.000000	0.0
95%	1.000000	12.000000	5.000000	0.0
max	44.000000	198.000000	71.000000	0.0

Figura 8: Descrição dos dados - pós limpeza de outliers de reaberturas

Tabela 5: G5 x G1.4

DataFrame/Parâmetros	Commits	Comentários (95% dos dados)	Usuários Envolvidos (95% dos dados)
G5 - Outliers de Reaberturas	100% são SBF (G2)	31	11
G1.4 - Dados Gerais - Sem Outliers de Reaberturas	95% possui até 1	12	5

de 6 a 9 contém os valores obtidos para os intervalos de confiança calculados.

Analisando os dados, nota–se que ao se comparar os intervalos do conjunto de dados G1.X e seus parâmetros (*commits*, comentários, usuários envolvidos e reaberturas) com os demais, não há interseção, ou seja, os *outliers* se diferem dos dados gerais sem outliers. Caso houvesse interseção e ela fosse forte, não seria possível afirmar nada sobre os dados.

Pode-se notar que ao calcular o intervalo de confiança para G1.1 em relação ao parâmetro reabertura foi obtido como resultado *nan*. Esta sigla significa que o valor encontrado não é um número (not a number). Isto acontece por que no processo de limpeza dos dados todas as reaberturas foram retiradas dos dados gerais, uma vez que todas as reabertura ocorreram em *issues* que possuem mais de um *commit* associado.

Para reforçar esta afirmação, a Tabela **??** contém os p-valores encontrados. Para calculá-los foi utilizado o grupo G1.X e o grupo de *outlier* correspondente. Por exemplo, para encontrar o p-valor de *G2 (outlier de Commits (SBF))* foi utilizado o grupo *G1.1: (Dados Gerais - Sem outliers de Commmits)*, e assim sucessivamente. Devido a eles serem abaixo de 0,05 (5%), significa que a chance de pegarmos uma outra amostra nos dados similar a esta amostra do *outlier* é baixa e de fato esses conjuntos de dados (*outliers*) se diferem dos demais dados da amostra geral.

Através dos resultados obtidos dos intervalos de confiança e valores p, a hipótese nula proposta na Seção 5.2 pode ser rejeitada e, consequentemente, a hipótese alternativa é aceita. Esta diferença de comportamentos dos dados nos grupos de *outliers* (G2 a G5) mostram que os parâmetros selecionados podem ter uma relação forte com o focos retrabalho. Os resultados encontrados por An, Khomh e Adams [1] mostram que reaberturas são um subconjunto de *Supplementary Bug Fixes*. As amostras analisadas neste trabalho, mostram que todas as *issues* com reaberturas são SBF. *Supplementary Bug Fixes* também podem ser resultado da falta de qualidade das mensagens trocadas ou por deficiência nas especificações, pois mesmo havendo muita discussão, 53.75% possuem de 1 a 6 comentários e 24.49% possui mais de 6 comentários, em torno da *issue*, foram necessário diversas mudanças (*commits* suplementares). Os grupos G3 e G4 também devem ser considerados ao se analisar focos de retrabalho, pois também apresentaram comportamento diferente em relação aos dados gerais. Nesses dois grupos, foi observado que apesar das *issues* terem muita discussão, elas não convergiram para código, ou seja, houve um desperdício dos recursos uma vez que o principal resultado esperado não foi obtido.

6 CONCLUSÃO

Este trabalho apresentou uma abordagem de análise de focos de retrabalho baseada nos parâmetros *número de commits*, *número de comentários*, *número de desenvolvedores* e *número de reaberturas*. Os resultados obtidos através das análises realizadas em 111575 *issues* de 17 projetos armazenados no *GitHub* mostraram que os parâmetros selecionados podem ter uma forte relação com focos de retrabalho.

Foram investigadas duas categorias de retrabalho relacionadas com as 3 fases do ciclo de vida da mudança. A primeira delas envolveu as duas primeiras fases do ciclo da mudança, Identificação da Issue e Avaliação da Issue , onde foi possível observar que apesar de algumas *issues* envolverem muita discussão e desenvolvedores, não se tem código resultante (*commits*), ou seja, houve um desperdício dos recursos envolvidos. Foi descoberto ao analisar G3 (*outlier de comentários*) e G4 (*outliers de usuários envolvidos*) que 72.01% e 84.34% das *issues* não possuem *commits associados*. A segunda categoria de *issues* analisada abrangeu a última fase do ciclo da mudança, Execução da Mudança, em que foram analisados dois grupos de *outliers*, G2 (*outliers de commits (SBF)*) e G5 (*outliers de reaberturas*). De forma similar aos resultados obtidos por An, Khomh e Adams

Tabela 6: Intervalos de Confiança - G1.1 e G2

DataFrame/Parâmetros	Comentários	Usuários	Reaberturas
G1.1: Dados Gerais - Sem Outliers de Commits	(3.291, 3.351)	(1.860, 1.883)	(nan, nan)
G2: Outliers de Commits (SBF)	(4.682, 5.671)	(2.086, 2.435)	(-0.167, 0.180)

Tabela 7: Intervalos de Confiança G1.2 e G3

DataFrame/Parâmetros	Commits	Usuários	Reaberturas
G1.2: Dados Gerais - Sem Outliers de Comentários	(0.430, 0.437)	(1.652, 1.669)	(0.0003, 0.0006)
G3: Outlier de Comentários	(0.313, 0.351)	(6.607, 6.892)	(-0.135, 0.149)

Tabela 8: Intervalos de Confiança G1.3 e G4

DataFrame/Parâmetros	Commits	Comentários	Reaberturas
G1.3: Dados Gerais - Sem Outliers de Usuários Envolvidos	(0.434, 0.441)	(2.794, 2.837)	(0.0004, 0.0008)
G4 - Outliers de Usuários Envolvidos	(0.183, 0.222)	(17.190, 17.990)	(0.002, 0.006)

Tabela 9: Intervalos de Confiança G1.4 e G5

DataFrame/Parâmetros	Commits	Comentários	Usuários
G1.4: Dados Gerais - Sem Outliers de Reaberturas	(0.424, 0.431)	(3.308, 3.369)	(1.863, 1.887)
G5: Outliers de Reaberturas	(2.258, 3.074)	(9.040, 13.812)	(3.501, 5.272)

[1], foi encontrado que as reaberturas são um subconjunto dos *Supplementary Bug Fixes*. No entanto, para as amostras analisadas, além de todas reaberturas encontradas serem também SBF, o percentual obtido foi mais significativo (100%). Além disso, o percentual de SBF com reaberturas encontrado neste trabalho foi significativamente inferior (94.49%) aqueles encontrados por An, Khomh e Adams (21.6 a 33.8%). Uma das possíveis causas de divergência dos resultados pode ser o fato da quantidade, tamanho e natureza dos projetos analisados serem diferentes, ou ainda em virtude da relação entre SBF e reaberturas ainda requerer novas análises.

Para comprovar a diferença entre os dados das amostras de *outliers* e os dados limpos, foi utilizado o teste de hipóteses. Através da reprovação da hipótese nula e aceitação da hipótese alternativa, proposta no teste de hipótese, pode-se comprovar que há diferença no comportamento dos dados. Essa diferença indica que os parâmetros selecionados podem ter uma forte relação com os focos de retrabalho.

Trabalhos futuros podem tentar adicionar às análises parâmetro tais como: origem e perfil do desenvolvedor (colaborador ou contribuidor); tipo do projeto (mantido por empresa ou comunidade); entre outros para se encontrar novos focos de retrabalho ou ainda para explicar melhor os parâmetros já selecionados neste trabalho.

REFERENCES

[1] L. An, F. Khomh, and B. Adams. 2014. Supplementary Bug Fixes vs. Re-opened Bugs. In *2014 IEEE 14th International Working Conference on Source Code Analysis and Manipulation*. 205–214. https://doi.org/10.1109/SCAM.2014.29

[2] Lars-Ola Damm, Lars Lundberg, and Claes Wohlin. 2008. A model for software rework reduction through a combination of anomaly metrics. *Journal of Systems and Software* 81, 11 (2008), 1968 – 1982. https://doi.org/10.1016/j.jss.2008.01.017

[3] Gláucia Braga e Silva. 2013. *COLMEIA - Um analisador de colaborações baseado em metricas aplicaveis a informações semanticamente integradas em um ambiente de GCS*. Ph.D. Dissertation. Instituto Tecnológico de Aeronáutica.

[4] Rodrigo Rocha Gomas e Souza. 2015. *Inappropriate Software Changes: Rejection and Rework*. Ph.D. Dissertation. Universidade Federal da Bahia.

[5] R. E. Fairley and M. J. Willshire. 2005. Iterative rework: the good, the bad, and the ugly. *Computer* 38, 9 (Sept 2005), 34–41. https://doi.org/10.1109/MC.2005.303

[6] IEEE 2010. In *Systems and Software engineering - Vocabulary*. ISO, New York, NY, USA, 308.

[7] E. Morozoff. 2010. Using a Line of Code Metric to Understand Software Rework. *IEEE Software* 27, 1 (Jan 2010), 72–77. https://doi.org/10.1109/MS.2009.160

[8] G Ramdoo, Vimla Huzooree. 2015. Strategies to reduce rework in software development on an organisation in Mauritius. *International Journal of Software Engineering & Applications* (2015), 9–20. https://doi.org/10.5121/ijsea.2015.6502

[9] Asra Khalid Ayesha Raana M. Nadeem Majeed Sobia Zahra, Ambreen Nazir. 2014. Performing Inquisitive Study of PM Traits Desirable for Project Progress. *International Journal of Modern Education and Computer Science (IJMECS)* 6, 2 (Feb. 2014), 41–47. https://doi.org/10.5815/ijmecs.2014.02.06

[10] Ian Sommerville. 2011. *Engenharia de Software* (9nd. ed.). PEARSON.

[11] Rúbio Thiago R. P. M. and Carlos A.S.J Gulo. 2015. Characterizing Developer's Rework on GitHub Open Source Projects. In *Proceedings of the 10th Doctoral Symposium in Informatics Engineering*, A. Augusto Sousa and Eugénio Oliveira (Eds.). Doctoral Symposium in Informatics Engineering - DSIE'15, 70–81.

[12] Charles Wheelan. 2016. *Estatística: O que é, para que serve, como funciona*. Zahar.