A refatoração é sempre um bom ovo? Explorando a interconexão Entre Bugs e Refatorações

Amirreza Bagheri
Universidade de Szeged
Hungria
bagheri@inf.u-szeged.hu

3pWHU+HJHGÿV
Universidade de Szeged
Hungria
hpeter@inf.u-szeged.hu

ABSTRATO

A correção de bugs e a refatoração de código são duas ações de manutenção distintas com objetivos diferentes. Embora a correção de bugs seja uma mudança corretiva que elimina um defeito do programa, a refatoração visa melhorar a qualidade interna (ou seja, a capacidade de manutenção) de um sistema de software sem alterar sua funcionalidade. As melhores práticas e a intuição comum sugerem que essas ações de código não devem ser misturadas em uma única alteração de código. Além disso, como a refatoração visa melhorar a qualidade sem alterações funcionais, esperamos que as alterações de código de refatoração não sejam fontes de bugs.
No entanto, estudos empíricos mostram que nenhuma das hipóteses acima é necessariamente verdadeira na prática. Neste artigo, investigamos empiricamente a interconexão entre alterações de código relacionadas a bugs e refatoração usando o conjunto de dados SmartSHARK. Nosso objetivo é explorar a frequência com que correções de bugs e refatorações ocorrem em um único commit (mudanças emaranhadas) e se as próprias mudanças de refatoração podem induzir bugs no sistema. Descobrimos que não é incomum haver commits emaranhados de correções de bugs e refatorações; 21% dos commits de correção de bugs incluem pelo menos um tipo de refatoração em média. O que é ainda mais chocante é que 54% dos commits indutores de bugs também contêm mudanças de refatoração de código. Por exemplo, 10% (652 ocorrências) das refatorações Change Variable Type no conjunto de dados aparecem em commits indutores de bugs que compõem 7,9% do total de commits indutores.

PALAVRAS-CHAVE

Erros induzindo commits, refatoração, mudanças de código emaranhadas, análise empírica

Formato de referência ACM:

$PLUUH]D%DJKHULDQG3pWHU+HJHGÿV 2022. A refatoração é sempre um bom ovo? Explorando a interconexão entre bugs e refatorações. In *Proceedings of MSR'22: Proceedings of the 19th International Conference on Mining Software Repositories (MSR 2022). ACM, Nova York, NY, EUA, 5 páginas.*

1. INTRODUÇÃO

Corrigir defeitos de software e melhorar a estrutura do código com refatoração são duas das ações de manutenção de software mais comuns. Eles são inerentemente diferentes na natureza. Uma correção de bug é uma modificação de código corretiva que corrige um falha no programa. Os desenvolvedores corrigem o comportamento indesejável alterando o código, banco de dados ou configuração, entre outras coisas. O método que eles usam para corrigir o bug será determinado pelo tipo de bug.
A refatoração [6], por outro lado, é uma modificação de código visando a melhoria da qualidade interna do software sem alterar seu comportamento funcional. Como essas atividades típicas de desenvolvimento de software são muito diferentes em sua natureza, os pesquisadores as estudaram extensivamente separadamente.

Ter bibliotecas de alta qualidade e grande escala de bugs validados e seus patches concisos coletados de aplicativos do mundo real são cruciais para estudá-los. Por um lado, bugs/patches reais são necessários para um exame completo de uma variedade de métodos automáticos ou semi-automáticos para detectar programas de software problemáticos, para encontrar declarações incorretas [7], [8] e corrigir aplicativos incorretos [9], [10]. Espera-se que esses métodos funcionem em situações do mundo real. Como resultado, antes que tais abordagens possam ser amplamente utilizadas em campo, elas devem ser avaliadas com um número significativo de bugs/patches reais de aplicações do mundo real [11]. Bugs e correções reais, por outro lado, podem fornecer inspiração para encontrar, localizar e reparar falhas de software. Os pesquisadores podem, por exemplo, estabelecer quais tipos de instruções são mais propensas a erros avaliando defeitos genuínos e, em seguida, tentar reparar essas instruções primeiro durante o reparo autônomo do programa [12]. Os padrões de reparo comuns aprendidos com patches escritos por humanos são outro bom exemplo. O uso de tais padrões melhorou muito o desempenho do reparo automático do programa.

[13]. Finalmente, estatísticas e métodos baseados em aprendizado para reparo autônomo de software [14] e identificação de bugs [15] também dependem de muitos bugs/patches reais.

A comunidade de pesquisa também passou muito tempo analisando a refatoração de software. Estudos observacionais investigaram por que e como os desenvolvedores realizam a refatoração [16], [17], [18], [19], [20] quais refatorações estão conectadas aos indicadores de desempenho do aplicativo [21], [22], [23], [24]], produtividade dos desenvolvedores [25] e como a refatoração se relaciona com outras tarefas de desenvolvimento[26] .

Como as refatorações são ações de melhoria de qualidade que não alteram a funcionalidade e as correções de bugs são alterações direcionadas para corrigir falhas funcionais, seria de se esperar que essas atividades fossem independentes umas das outras.
No entanto, apesar da intuição, alguns pesquisadores começaram a estudar a relação entre bugs e atividades de refatoração. Curiosamente, de acordo com alguns pesquisadores [27], os desenvolvedores geralmente ficam apreensivos com os esforços de refatoração, pois podem introduzir defeitos. Vários estudos investigaram a relação entre refatoração [28] e bugs, analisando repositórios de software para ver o quanto as atividades de refatoração introduzem bugs [29]. Weibgerber e Diehl [30] investigaram a

relação entre as ações de refatoração e o número de relatórios de bugs abertos nos dias seguintes e não encontrou nenhuma ligação significativa.

Neste trabalho, aproveitamos a riqueza de dados relacionados a bugs e atividades de refatoração registrados no conjunto de dados SmartSHARK [1] para investigar empiricamente a interconexão entre bugs e refatorações. Alinhados com trabalhos anteriores, descobrimos que não é incomum haver commits emaranhados de correções de bugs e refatorações; 21% dos commits de correção de bugs incluem pelo menos um tipo de refatoração em média. O que é ainda mais chocante é que 54% dos erros

MSR '22, 23 a 24 de maio de 2022, Pittsburgh, PA, EUA ©
2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9303-4/22/05…$15,00 https://
doi.org/10.1145/3524842.3528034

induzir commits também contém mudanças de refatoração de código. Também identificamos os tipos de refatoração que aparecem com mais frequência em commits indutores de bugs.

2 PROJETO DE ESTUDO

Para explorar as interconexões entre bugs e refatorações, projetamos um estudo baseado no conjunto de dados SmartSHARK [1] de repositórios de software Java [2], que contém dados para 96 projetos, em particular as atividades de refatoração e correção de bugs, links validados manualmente entre commits e problemas de bugs, bem como o tipo de problemas [4], e rótulos de linha validados manualmente que marcam quais mudanças contribuíram para uma correção de bug [5].

O SmartSHARK não é apenas um conjunto de dados, mas uma plataforma para mineração de repositório de software replicável e reproduzível, um conjunto de dados que combina informações detalhadas do sistema de controle de versão com dados de rastreamento de problemas, dados de pull request do GitHub e dados do Travis CI. Todos os dados dessa base de dados também possuem links para as diversas fontes de informação. Abordamos as seguintes questões de pesquisa com a ajuda do SmartSHARK: ***RQ1. Quão comum é que um commit de correção de bug também contenha mudanças de refatoração?***

Nossa hipótese é que as correções de bugs devem ser mudanças independentes, não incluindo nenhum outro tipo de modificação de código. No entanto, estudos anteriores [27], [28] sugerem que, na prática, os commits geralmente contêm mudanças de código emaranhadas. Portanto, investigamos o quão comum é que os desenvolvedores executem ações de refatoração emaranhadas com correções de bugs.

RQ2. As operações de refatoração aparecem em modificações de código induzindo bugs?

Investigamos se as atividades de refatoração podem levar à introdução de bugs no sistema (ou seja, podemos detectar uma atividade de correção de bugs no código refatorado posteriormente no histórico de commits). Para isso, analisamos se os commits marcados como 'indutores de bug' no dataset também contêm ações de refatoração ou não.

RQ3. Quais são os tipos de refatoração mais comuns que aparecem em commits indutores de bugs?

Por fim, se descobrirmos que as alterações indutoras de erros também podem conter modificações de código de refatoração, exploramos quais são os tipos mais comuns de refatoração que observamos. Isso pode nos ajudar a entender quais são os tipos de refatoração mais 'perigosos' onde os desenvolvedores precisam prestar atenção especial para não introduzir bugs junto com as modificações.

3 MINERAÇÃO SMARTSHARK

Para realizar o estudo e responder às nossas questões de pesquisa, analisamos o histórico de alterações de 96 projetos armazenados no conjunto de dados SmartSHARK versão 2.2. É fundamental em nosso estudo identificar os commits de correção de bugs e aqueles que fazem referência ao id do problema resolvido pelo commit. Com relação ao primeiro ponto (ou seja, rótulos para bugs), cada commit no SmartSHARK tem um conjunto de rótulos indicando se aquele commit é uma correção de bug ou não, que são automaticamente inferidos por heurística ou confirmados por validação manual. Em termos do segundo problema, ter um link explícito entre commits e defeitos nos permite identificar os commits de correção de bugs que precisamos para nossa pesquisa. Para encontrar commits contendo operações de refatoração, usamos o SmartSHARK's

Resultados da ferramenta de detecção RMiner. Espera-se que a precisão e a recuperação do Rminer [3] sejam de 98% e 87%, respectivamente. Para encontrar atualizações indutoras de correção, procuramos ações de arquivo relacionadas aos commits com um sinalizador 'induzindo'. Todos os scripts de mineração e dados coletados estão disponíveis online.1

4 RESULTADOS DO ESTUDO

RQ1. Quão comum é que um commit de correção de bug também contenha mudanças de refatoração?

Refatorações são modificações de código-fonte que preservam o comportamento, de acordo com Fowler [6]. O objetivo fundamental da refatoração é aumentar a capacidade de manutenção ou compreensão, bem como minimizar a pegada do código, se necessário. Aqui, analisamos se a atividade de refatoração pode ser desencadeada por um bug no código, analisando os commits de correção de bugs. Descobrimos que 41 de 96 projetos validaram commits de correção de bugs que também contêm atividades de refatoração. Além disso, para esses projetos, um grande parte dos commits de correção de bugs também foram rotulados como commits de refatoração. Contamos com o rótulo 'validated_bugfix' armazenado no SmartSHARK para cada confirmação para identificar. Para descobrir se esse commit contém ações de refatoração, observamos as entradas na coleção 'refactoring' referenciando o mesmo id de commit. Se uma entrada de refatoração apontando para o mesmo id de commit tivesse o valor 'rMiner' em seu campo 'detection_tool', nós identificamos esse commit de correção de bug afetado pela refatoração também. Encontramos um total de 2.345 commits de correção de bugs que também incluem refatorações no conjunto de dados SmartSHARK.

Projeto	NRB	RNB	RB	R% 0.23 0.27 0.41 0.19 0.14
anti-hera	440	447	128	0.14
archiva	396	1107	148	0.19
calcite	250	908	177	0.15
cayenne	686	1479	164	0.20
commons-bcel	42	130	7	0.25
commons-beanutils	51	161	8	0.17
commons-codec	48	161	11	0.15
commons-collections	75	572	13	0.23
commons-compress	165	382	41	0.18
commons-configuration	182	759	61	0.26
commons-dbcp commons-	88	243	18	0.33
digester commons-imaging	22	216	4	0.12
commons-io commons-jcs	20	211	6	0.20
commons-jexl commons-lang	98	220	21	0.22
commons-math commons-net	53	288	19	0.31
commons-scxml commons-	109	331	54	0.05
validator commons-vfs	213	552	29	0.23
deltaspike diretório-fortress-	316	1075	80	0.24
core eagle falcon giraph gora	138	168	38	0.19
jspwiki knox kylin lens mahout	46	178	21	0.28
manifoldcf nutch opennlp	69	129	4	0.31
parquet-mr santuario-java	88	313	26	0.23
systemml tika wss4j **Total/**	165	413	52	0.11
Avg.	42	179	10	0.11
	94	211	36	0.20
	218	534	98	0.17
	109	329	32	0.32
	91	161	11	0.18
	209	904	25	0.17
	279	381	69	0.15
	1052	2241	212	0.14
	187	480	89	0.30
	269	756	59	0.28
	5755	789	116	0.32
	5361	403	83	0.20
		283	20	0.25
		492	36	
		466	27	
		1427	97	
		686	134	
		764	61	
	8539	**21929**	**2345**	**0,21**

Tabela 1. Nenhum commit de refatoração e correção de bug (NRB); Refactoring e commits No Bug-fix (RNB); Confirmações de refatoração e correção de bugs (RB); R% = proporção RB/(RB+NRB) de commits RB de todos os commits de correção de bugs

A Tabela 1 apresenta os resultados detalhados para os 41 projetos que encontramos commits de correção de bugs emaranhados com ações de refatoração. Para o resto dos projetos, não encontramos nenhum commit de correção de bugs ou apenas correções de bugs que não incluíam nenhuma operação de refatoração (portanto, a contagem de RB é 0). A primeira coluna (NRB) na tabela mostra o número de commits de correção de bugs que não incluem refatorações, a segunda coluna (RNB) mostra o número de commits identificados como refatorações, mas não correções de bugs, a terceira coluna é o número de commits onde a correção de bugs e a refatoração ocorrem, e a última coluna mostra o

[1] https://doi.org/10.5281/zenodo.6381329

proporção de commits de RB em comparação com todos os commits de correção de bugs. Como pode ser visto, 21% dos commits de correção de bugs incluem pelo menos um tipo de refatoração em média. Portanto, não é incomum realizar refatorações junto com a correção de bugs. Esta proporção é a mais baixa (5%) para o projeto 'commons-validator', enquanto a mais alta (41%) para o projeto 'calcita'. No entanto, isso não significa que a maioria das atividades de refatoração aconteça junto com correções de bugs. Existem 21.929 commits contendo refatoração, mas não rotulados como correção de bug. O número de commits de refatoração que também são commits de correção de bugs é 2345, que é um pouco mais de 10% de todos os commits de refatoração.

Resposta a RQ1. Não é incomum que correções de bugs e refatorações de código ocorram simultaneamente no mesmo commit. Descobrimos que 41 dos 96 projetos tinham esses commits. Observamos a maior proporção desses commits em comparação com o número total de commits de correção de bugs para o projeto 'calcite', onde 41% de todas as mudanças no código de remoção de bugs continham pelo menos uma operação de refatoração também. No entanto, as refatorações normalmente não ocorrem emaranhadas com correções de bugs, pois apenas 10% de todas as ações de refatoração nesses projetos foram identificadas em commits de correção de bugs.

RQ2. As operações de refatoração aparecem em modificações de código induzindo bugs?

Em teoria, a refatoração é descrita como a execução de ações simples de tal forma que são "improváveis de dar errado" e gerar erros. Para investigar empiricamente essa hipótese, estudamos as mudanças indutoras de correção em commits que se sobrepõem a ações de refatoração (ou seja, busca por commits que contenham mudanças indutoras de bugs e incluam pelo menos uma operação de refatoração também).
Ressaltamos aqui que, novamente, observamos apenas a co-ocorrência de alterações e refatorações indutoras de erros. Uma refatoração em si só induz um bug se uma linha afetada pela refatoração também for modificada em um commit de correção de bug posteriormente no histórico do projeto. No entanto, uma refatoração ainda pode influenciar a introdução de um bug, mesmo que não toque diretamente em uma linha alterada na correção do bug. Portanto, é muito difícil decidir se uma refatoração contribui para a introdução de bugs ou simplesmente ocorre concomitantemente com mudanças indutoras de bugs. Ressaltamos que aqui apenas analisamos se as operações de refatoração aparecem em commits indutores de bugs ou não, mas não rastreamos as linhas modificadas exatas de volta para correções de bugs. No entanto, é em si uma questão empírica interessante se as operações de refatoração estão associadas a mudanças indutoras de bugs ou não.

Usamos os commits de correção de bugs como um começo para ir mais fundo e ver o que está acontecendo nos arquivos afetados por essas alterações de código. Para isso, podemos localizar commits indutores de bugs dentro das entradas FileAction vinculadas aos commits.
Em nosso estudo, usamos os commits indutores rotulados como 'JLMIV+' (Jira Links Manual(JLM), Issue Validation(IV), apenas arquivos java(+), pular comentários e espaços vazios em culpa(+)) que também tinham ' szz_type' para o campo 'inducing' e omitiu os rótulos 'hard_suspect'. Em seguida, investigamos as entradas do FileAction para 'change_file_action_id' e pesquisamos os problemas para problemas vinculados nesses commits e se eles tivessem um tipo de problema rotulado como 'bug' e nas mesmas refatorações de commit foram detectadas, nós os coletamos.

Esses commits são aqueles que induzem um bug no sistema e também contêm operações de refatoração. Um dos projetos ('commons-imaging') não teve nenhuma alteração indutora de bugs, portanto, temos dados para 40 projetos nesta análise.

Em nossas descobertas exibidas na Tabela 2, descobrimos que mais de 54% dos commits indutores de bugs nos projetos selecionados também contêm ações de refatoração, em média. As porcentagens do projeto variam de 20%
(projeto 'commons-validator') até 71% (projeto 'calcite') de todos os commits indutores de bugs em um projeto, o que é muito significativo. As operações de refatoração ocorrem com bastante frequência em alterações de código que levam à introdução de bugs. Curiosamente, os mesmos dois projetos são os dois extremos como no caso do RQ1. Uma vez que 'commons-validator' contém o menor número de commits de refatoração (133) entre os 40 projetos, não é surpreendente que este seja o projeto onde a proporção de commits de refatoração co-ocorrendo com correções de bugs e mudanças indutoras de bugs também são o mais baixo. No caso de 'calcita', no entanto, observamos um número alto (1085) de commits de refatoração, mas não é o mais alto. No entanto, ele contém a maior proporção de correções de bugs e commits indutores de bugs emaranhados com refatorações entre os 40 projetos. 71% de todas as mudanças indutoras de bugs também incluem operações de refatoração.

Projeto	#Indução de	RI	R%
anti-hera	**insetos**	169	0.53
archiva	313 444	225	0.50
calcite	408 522 51	292	0.71
cayenne	36 44 52	317	0.60
commons-bcel	151 150	13	0.25
commons-beanutils	63 20 67	17	0.47
commons-codec commons-	128 200	16	0.36
collections	171 284	34	0.65
commons-compress	110 54 35	68	0.45
commons-configuration	94 153 63	76	0.50
commons-dbcp commons-	107 282	28	0.44
digester commons-io commons-	142 37	6	0.30
jcs commons-jexl commons-	244 188	26	0.38
lang commons-math commons-	879 311	88	0.68
net commons-scxml commons-	341 440	117	0.58
validator commons-vfs diretório	270 99 124	52	0.30
deltaspike-fortress-core eagle	96 490 383	128	0.45
falcon giraph gora jspwiki knox	187	22	0.20
kylin lens mahout manifoldcf		29	0.53
nutch opennlp parquet-mr		7	0.20
santuario-java systemml tika		44	0.46
wss4j **Total/Avg.**		86	0.56
		33	0.52
		71	0.66
		180	0.63
		104	0.73
		20	0.54
		82	0.33
		106	0.56
		525	0.59
		191	0.61
		213	0.62
		218	0.49
		125	0.46
		41	0.41
		82	0.66
		60	0.62
			0.62
			0.43
			0.56
	8233	**4490**	**0,54**

Tabela 2. Número total de commits indutores de bugs (#Bug-Induce), commits indutores de bugs contendo refatoração também (RI) e sua proporção (R%)

Resposta a RQ2. A presença de refatoração de código é ainda mais significativa em commits indutores de bugs do que em commits de correção de bugs. Mais da metade das mudanças que induzem bugs também contêm operações de refatoração, em média. Para 'commons-validator' observamos a menor proporção (20%), enquanto para 'calcite' a maior (71%) de commits indutores de bugs emaranhados com refatorações.

RQ3. Quais são os tipos de refatoração mais comuns que aparecem em commits indutores de bugs?

Como descobrimos que uma parte significativa (54% em média) dos commits indutores de bugs também contém operações de refatoração, usamos o conjunto de dados SmartSHARK mais uma vez para coletar os tipos reais de refatorações que aparecem em tais commits. Por causa deste RQ, avaliamos apenas o caso em que a refatoração se sobrepõe à indução de bug no nível de confirmação. A Tabela 3 lista os diferentes tipos de refatoração e mostra o número de commits incluindo pelo menos uma operação de refatoração (primeira coluna, #Refact), o número de commits indutores de bugs com esse tipo de refatoração (segunda coluna, #Bug-Induce) , a proporção de commits indutores de erros contendo esse tipo de refatoração em comparação com o número total de commits contendo essa refatoração (terceira coluna, R1%) e a proporção de commits indutores de erros em comparação com o número total de commits indutores de erros com essa operação de refatoração no conjunto de dados completo (quarta coluna, R2%).
Pequenas refatorações que alteram tipos ou renomeiam atributos/tipos (alterar variável

type, change return type, renomear variável/atributo), bem como extrair partes de código (extrair método/atributo/variável/classe) têm as maiores chances de aparecer em mudanças de código indutoras de bugs (maior R2%). Por exemplo, a refatoração Change Variable Type ocorre em 7,92% de todos os commits indutores de bugs. No entanto, exatamente essas são as refatorações com mais ocorrências no conjunto de dados. Observando os valores de R1%, podemos ver que Extract Subclass está no topo com 33%. Isso significa que um terço dessa operação de refatoração acontece em alterações de código que induzem bugs.

Digite	#Refato	#Indução de Bug	R1%	R2%	0,10
change_variable_type	6475	652	0,0792	0,09	0,0551
extract_method	5257	454			
change_return_type	3529	338	0,10	0,0411	
extract_attribute	3306	331	0,10	0,0402	0,08
rename_variable	2844	241	0,0293		
rename_attribute	1724	213	0,12	0,0259	
rename_parameter	2200	208	0,09	0,0253	
extract_and_move_meth d	1228	206	0,17	0,025	
move_method	1534	206	0,13	0,0250	
extract_variable	1697	199	0,12	0,0242	0,24
extract_class	680	162	0,0197		
rename_method	2406	143	0,06	0,0174	
extract_superclass	720	136	0,19	0,0165	
rename_class	1148	127	0,11	0,0154	
move_attribute	1224	116	0,09	0,0141	
inline_method	846	111	0,13	0,0135	
parametrize_variable	614	102	0,17	0,0124	
move_class inline_variable	1664	87	0,05	0,0106	
pull_up_method	734	57	0,08	0,0069	
replace_variable_with_at	705	57	0,08	0,0069	
	479	46	0,10	0,0056	
extract_subclass	123	41	0,33	0,0050	
pull_up_attribute	467	39	0,08	0,0047	0,17
push_down_method	230	38	0,0046		
push_down_attribute	170	36	0,21	0,0044	
move_and_rename_class	479	32	0,07	0,0039	
extract_interface	206	28	0,14	0,0034	
merge_variable	159	23	0,14	0,0028	
merge_parameter split_attribute	127	21	0,17	0,0026	0,21
merge_attribute split_variable	58	12	0,0015		
move_and_rename_attri	48	8	0,17	0,001	
	48	7	0,15	0,0009	
	18	5	0,28	0,0006	
replace_attribute	14	4	0,29	0,0005	
split_parameter	28	4	0,14	0,0005	

Tabela 3: commits indutores de bugs por tipos de refatoração

Resposta ao RQ3. Mudanças de tipo de variáveis e declarações de retorno, renomeação de atributos e variáveis, e extração de método ou atributo são as refatorações mais frequentes que aparecem em commits indutores de bugs. Além disso, 33% da refatoração da subclasse de extração aparece em commits indutores de bugs, embora tenha apenas algumas instâncias (123) no conjunto de dados. Por outro lado, instâncias da classe move raramente ocorrem em commits indutores de bugs (apenas 5% dos casos).

5. CONCLUSÕES

O objetivo deste estudo foi investigar a relação entre refatorações e correções de bugs ou mudanças de código indutoras de bugs. Como eles são atividades de manutenção de código completamente diferentes, nossa hipótese era que elas não ocorressem dentro do mesmo commit. No entanto, trabalhos anteriores já apontavam que isso pode não ser o caso na prática.

Usando os ricos dados disponíveis no conjunto de dados SmartSHARK, conseguimos conectar atividades de refatoração a commits de correção e indução de bugs. Para identificar as operações de refatoração, contamos com os resultados da ferramenta RMiner incluída no conjunto de dados, uma técnica de detecção altamente precisa com precisão relatada de 98% e recall de 87%. Para localizar commits indutores de bugs, usamos os dados produzidos pelo algoritmo SZZ, que é utilizado para detectar alterações indutoras de correção no SmartSHARK. Reconhecemos que a validade de nossos resultados depende da precisão dessas ferramentas/algoritmos e dos dados que eles produzem. No entanto, os muitos trabalhos de alta qualidade que contam com essas ferramentas e dados aumentam a confiança nos resultados apresentados.

Descobrimos que não é incomum ter operações de refatoração emaranhadas com correções de bugs em um único commit. Isso sugere que os desenvolvedores na prática realizam melhorias na estrutura do código ao encontrar e corrigir um defeito de software. Em 41 dos 96 projetos, descobrimos que, em média, 21% dos commits de correção de bugs também contêm refatorações. Embora a porcentagem mais alta tenha sido de 41%, as refatorações normalmente não ocorrem emaranhadas com correções de bugs, pois apenas 10% de todas as ações de refatoração nesses projetos foram identificadas em commits de correção de bugs.

A presença de refatorações de código foi ainda mais significativa em commits indutores de bugs, onde 54% desses commits continham pelo menos uma operação de refatoração. Devemos observar, no entanto, que analisamos apenas a co-ocorrência de refatorações em alterações indutoras de erros, não que as linhas refatoradas tenham contribuído diretamente para o defeito introduzido. Portanto, uma refatoração concomitante pode não ser a causa raiz dos defeitos.

No entanto, esse alto número de refatorações no código indutor de bugs é alarmante. Esse resultado pode disparar um alerta para os praticantes, apontando que a refatoração nem sempre pode preservar o comportamento na prática, portanto, os desenvolvedores devem estar preparados com técnicas adequadas de verificação e validação para mitigar os riscos potenciais causados pelas refatorações.

Nós até identificamos que mudanças de tipo de variáveis e declarações de retorno, renomear atributos e variáveis, e extrair método ou atributo são as refatorações mais frequentes que aparecem em commits indutores de bugs. Além disso, 33% da refatoração da subclasse de extração aparece em commits indutores de bugs, embora tenha apenas algumas instâncias no conjunto de dados. Por outro lado, instâncias da classe move raramente ocorrem em commits indutores de bugs (apenas 5% dos casos).

Neste estudo, fornecemos dados quantitativos que mostram a interconexão de refatorações e correções de bugs, bem como refatorações e alterações de código indutoras de erros. No entanto, mais estudos qualitativos são necessários para descobrir a relação precisa entre eles. No futuro, planejamos realizar essa análise nos dados do SmartSHARK que extraímos.

RECONHECIMENTO

O trabalho apresentado foi realizado dentro do Projeto SETIT (2018-1.2.1-NKP-2018-00004). Projeto Não. 2018-1.2.1-NKP-2018-00004 foi implementado com o apoio do Fundo Nacional de Pesquisa, Desenvolvimento e Inovação da Hungria, financiado no âmbito do 2018-1.2.1-Esquema de financiamento do NKP.
)XUWKHUPRUH 3pWHU+JHGÿVZDVVXSSRUWHGE\WKH%RO\DL-iQRV6FKRDUVKLS da Academia Húngara de Ciências e o ÚNKP-21-5-SZTE-570 Novo Programa Nacional de Excelência do Ministério da Inovação e Tecnologia.

REFERÊNCIAS

[1] A. Trautsch, F. Trautsch, S. Herbold, “MSR Mining Challenge: The SmartSHARK Repository Data,” Proceedings of the International Conference on Mining Software Repositories (MSR 2022), 2022 [2] A. Trautsch, F. Trautsch, S. Herbold, B. Ledel e J. Grabowski, “O ecossistema smartshark para mineração de repositório de software”, em Proc. do Internacional 2020. Conf. Softw. Eng. - Faixa de Demonstrações, 2020.
[3] N. Tsantalis, M. Mansouri, LM Eshkevari, D. Mazinanian e D. Dig, “Detecção de refatoração precisa e eficiente em commit

history”, in Proceedings of the 40th International Conference on Software Engineering, ser. ICSE '18. Nova York, NY, EUA: ACM, 2018, pp. 483-494.

[4] S. Herbold, A. Trautsch e F. Trautsch, “Problemas com szz: Uma avaliação empírica do estado da prática da coleta de dados de previsão de defeitos”, 2019.

[5] S. Herbold, A. Trautsch, B. Ledel, A. Aghamohammadi, TA Ghaleb, KK Chahal, T. Bossenmaier, B. Nagaria, P. Makedonski, MN Ahmadabadi, K. Szabados, H. Spieker, M. Madeja, N. Hoy, V. LenarGX]]L 6 :DQJ * 5RGU ÕJXH]- Pérez, R. Colomo-Palacios, R. Verdecchia, P. Singh, Y. Qin, D. Chakroborti, W. Davis, V. Walunj, H. Wu, D. Marcílio, O. Alam, A. Aldaeej, I. Amit, B. Turhan, S. Eismann, A.-K.Wickert, I. Malavolta, M. Sulir, F. Fard, AZ Henley, S. Kourtzanidis, E. Tuzun, C. Treude, SM Shamasbi, I. Pashchenko, M. Wyrich , J. Davis, A. Serebrenik, E. Albrecht, E. U. Aktas, D. Strëuber e J. Erbel, “Large-scale manual validation of bug fixing commits: A finegrained analysis of tangling”, 2020.

[6] FOWLER, M., & BECK, K. (1999). Refatoração: melhorando a projeto de código existente. Reading, MA, Addison-Wesley

[7] WE Wong, R. Gao, Y. Li, R. Abreu e F. Wotawa, “Uma pesquisa sobre localização de falhas de software”, IEEE Transactions on Software Engineering, vol. 42, não. 8, pp. 707-740, 2016.

[8] J. Lee, D. Kim, TF Bissyand´e, W. Jung e Y. Le Traon, “Bench4BL: estudo de reprodutibilidade no desempenho de localização de bugs”, em Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis, 2018, págs. 61-72.

[9] D. Jeffrey, M. Feng, N. Gupta e R. Gupta, “BugFix: Uma ferramenta baseada em aprendizado para auxiliar desenvolvedores na correção de bugs”, em 2009 IEEE 17th International Conference on Program Comprehension. IEEE, 2009, pp. 70–79.

[10] B. Daniel, V. Jagannath, D. Dig e D. Marinov, “ReAssert: Sugerindo reparos para testes de unidade quebrados”, em 2009 IEEE/ACM International Conference on Automated Software Engineering. IEEE, 2009, pp. 433-444.

[11] R. Just, D. Jalali e MD Ernst, “Defects4J: A database of existing faults to enable managed testing studies for java programs”, in Proceedings of the 2014 International Symposium on Software Testing and Analysis, 2014, pp. 437-440.

[12] J. Xuan, M. Martinez, F. DeMarco, M. Clement, S. Marcote, T. Durieux, DL Berre e M. Monperrus, “Nopol: Reparação automática de erros de instrução condicional em programas Java”, IEEE Transactions on Software Engineering, vol. 43, nº. 01, pp. 34–55, janeiro de 2017.

[13] D. Kim, J. Nam, J. Song e S. Kim, “Geração automática de patches aprendida com patches escritos por humanos”, em 2013 35th International Conferência sobre Engenharia de Software (ICSE), 2013, pp. 802–811.

[14] Y. Xiong, J. Wang, R. Yan, J. Zhang, S. Han, G. Huang e L. Zhang, “Síntese de condição precisa para reparo de programa”, em Anais da 39ª Conferência Internacional de Engenharia de Software, ICSE 2017, Buenos Aires, Argentina, 20 a 28 de maio de 2017, S. Uchitel, A. Orso e MP Robillard, Eds . IEEE / ACM, 2017, pp. 416-426.

[15] H. Zhong, X. Wang e H. Mei, “Inferring bug signatures to detect real bugs”, IEEE Transactions on Software Engineering, pp. 1–1, 2020.

[16] Emerson Murphy-Hill, Chris Parnin e Andreaw P. Black. 2011. Como refatoramos e como sabemos disso. Transações em Engenharia de Software 38, 1 (2011), 5–18.

[17] Anthony Peruma, Mohamed Wiem Mkaouer, Michael J. Decker e Christian D. Newman. 2018. Uma investigação empírica de como e por que os desenvolvedores renomeiam identificadores. In Proceedings of the 2nd International Workshop on Refactoring (IWoR 2018). 26-33.

[18] Danilo Silva, Nikolaos Tsantalis e Marco Túlio Valente. 2016. Por que refatoramos? confissões de contribuidores do GitHub. In Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2016. 858–870.

[19] Carmine Vassallo, Giovanni Grano, Fabio Palomba, Harald Gall e Alberto Bacchelli. 2019. Uma exploração empírica em larga escala sobre atividades de refatoração em projetos de software de código aberto. Ciência da Programação de Computadores 180, 1 (2019), 1–15.

[20] Yi Wang. 2009. O que motiva os engenheiros de software a refatorar o código-fonte? Evidências de desenvolvedores profissionais. In Software Maintenance, 2009. ICSM 2009. IEEE International Conference on. 413-416.

[21] Mohammad Alshayeb. 2009. Investigação empírica do efeito de refatoração na qualidade de software. Tecnologia da Informação e Software 51, 9 (2009), 1319 – 1326.

[22] $OH[DQGHU &KiYH] ,VDEHOOD)HUUHLUD (GXDUGR)HUQDQGHV 'LHJR Cedrim e Alessandro Garcia. 2017. Como a refatoração afeta os atributos de qualidade interna?: Um estudo de vários projetos. Anais do 31º Simpósio Brasileiro de Engenharia de Software (SBES'17). 74-83.

[23] Konstantinos Stroggylos e Diomidis Spinellis. 2007. Refatoração– Melhora a qualidade do software? In Proceedings of the 5th International Workshop on Software Quality (WoSQ '07). IEEE Computer Society, Washington, DC, EUA,10–. [24] *iERU6]RNH *iERU$QWDO &VDED1DJ\ 5XGROI)HUHQF DQG7LERU Gyimóthy.2014. Corrigindo problemas de codificação em massa e seus efeitos na qualidade do software: vale a pena refatorar? Em Análise e Manipulação de Código Fonte (SCAM), 2014 IEEE 14th International Working Conference on. IEEE, 95-104.

[25] Raimund Moser, Pekka Abrahamsson, Witold Pedrycz, Alberto Sillitti e Giancarlo Succi. 2008. Equilibrando Agilidade e Formalismo em Engenharia de Software. Capítulo A Estudo de caso sobre o impacto da refatoração na qualidade e produtividade em uma equipe ágil, 252–266.

[26] Mehran Mahmoudi, Sarah Nadi e Nikolaos Tsantalis. 2019. As refatorações são culpadas? Um estudo empírico de refatorações em conflitos de mesclagem. Na 26ª Conferência Internacional do IEEE sobre Análise, Evolução e Reengenharia de Software, SANER 2019. 151–162.

[27] Gabriele Bavota, Bernardino De Carluccio, Andrea De Lucia, Massimiliano Di Penta, Rocco Oliveto e Orazio Strollo. 2012. Quando uma refatoração induz bugs? Um Estudo Empírico. Na 12th IEEE International Working Conference on Source Code Analysis and Manipulation, SCAM 2012, Riva del Garda, Itália, 23-24 de setembro de 2012. 104-113.

[28] Isabella Ferreira, Eduardo Fernandes, Diego Cedrim, Anderson Uchôa, Ana Carla Bibiano, Alessandro Garcia, João Lucas Correia, Filipe Santos, Gabriel Nunes, Caio Barbosa et al. 2018. O lado bugado da refatoração de código: Entendendo a relação entre refatorações e bugs. In Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings (ICSE'18). 406-407.

[29] Miryung Kim, Thomas Zimmermann e Nachiappan Nagappan. 2012. Um estudo de campo dos desafios e benefícios da refatoração. In Proceedings of the 20th International Symposium on Foundations of Software Engineering (Research Triangle Park, NC, EUA).

[30] Peter Weißgerber e Stephan Diehl. 2006. As refatorações são menos propensas a erros do que outras mudanças?. In Proceedings of the 2006 International Workshop on Mining Software Repositories, MSR 2006, Shanghai, China, 22-23 de maio de 2006. 112–118.