RAID: Tool Support for Refactoring-Aware Code Reviews

RAID: Ferramenta de suporte para revisões de código com reconhecimento de refatoração

Qualidade de Software

Aluno(a): Luis Gomes Damasceno Neto
Matrícula: 428223
Professor(a): Carla Ilane Moreira Bezerra

Autores

Marco Tulio Valente

ASERG Group,
Federal University of
Minas Gerais (UFMG),
Belo Horizonte, Brazil

Rodrigo Brito

ASERG Group,
Federal University of
Minas Gerais (UFMG),
Belo Horizonte, Brazil

RAID

https://github.com/rodrigo-brito/refactoring-aware-diff

A revisão de código é uma prática chave de desenvolvimento que contribui para melhorar a qualidade do software e promover o compartilhamento de conhecimento entre os desenvolvedores.

Porém, para revisar o código geralmente leva um bom tempo e requer uma análise detalhada e demorada, das diferenças textuais.

Mas detectar refatorações durante revisões de código é uma tarefa complexa, já que elas não são explicitamente representadas em diffs(comparação).

Para suprir essa necessidade, foi introduzido o *RAID (refatoring-aware and intelligent diffs)* , uma ferramenta de comparação inteligente e ciente de refatoração.

Além de sua arquitetura, foi implementado um plug-in do navegador Chrome que suporta essa solução.

Introdução

Esse artigo descreve os principais recursos e a arquitetura de uma ferramenta para dar suporte a revisões de código com reconhecimento de refatoração.

O objetivo é aliviar o esforço cognitivo associado às revisões de código, pois detecta automaticamente as operações de refatoração incluídas nas solicitações pull.

Além de oferecer suporte para detecção de refatoração, a ferramenta RAID instrumenta perfeitamente as ferramentas de comparação atuais com informações sobre as refatorações.

Como resultado, os revisores podem facilmente inspecionar as alterações realizadas no código de refatoração após a operação.

Esse artigo está divido em três seções principais

- Na seção 2 é apresentado os principais recursos do RAID e a interface baseada na Web.
- Na seção 3 é descrita a arquitetura interna do RAID, incluindo sua integração com ferramentas de terceiros, como github (ações e pull request) e navegador chrome.
- Na seção 4 foi documentado os resultados e lições aprendidas em um experimento de campo onde profissionais usaram o RAID.

RAID in a nutshell

Fig. 1. Diferença padrão incluindo uma refatoração Move Function (m1 é movido de A.java para B.java)

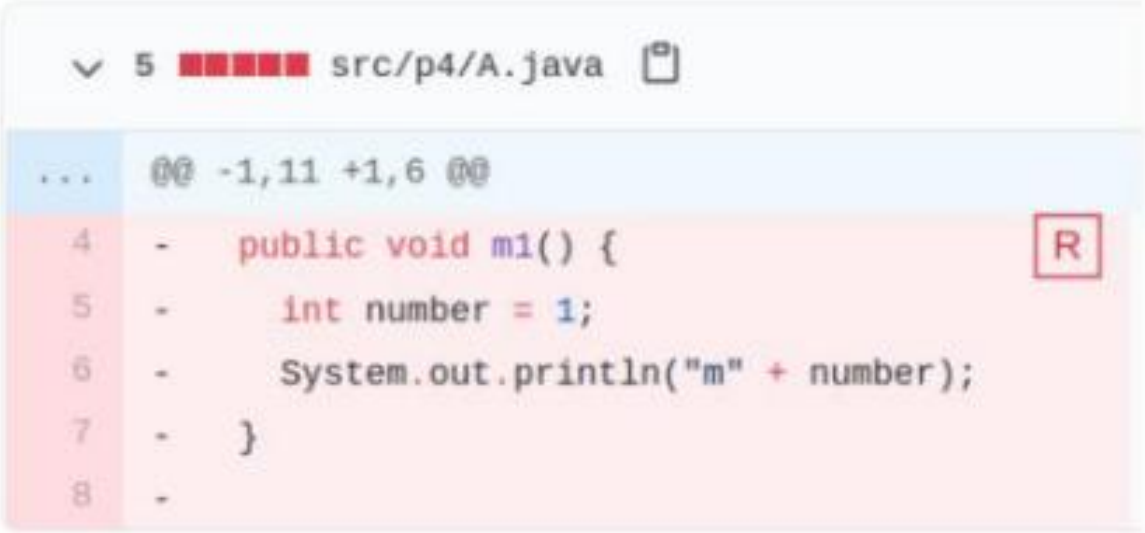


Fig. 2. Diff instrumentado por RAID (um botão "R" é adicionado ao diff indicando que a função é parte de uma refatoração)

Fig. 3. Exemplo de diff incluindo uma função Move, conforme apresentado pelo RAID (esta janela é aberta após clicar no botão "R" mostrado na Figura 2)

Fig. 4. Exemplo de janela documentando uma Função Extract, conforme apresentada pelo RAID. Podemos ver as mudanças de linhas no método original (topo) e também o código do método extraído (abaixo)

TABELA I

REFATORAMENTOS DETECTADOS PELO REFDIFF/RAID

Refatorações de linguagem	
Java	Mover, Extrair Função, Função Inline, Renomear, Alterar assinatura, puxar para cima, empurrar para baixo
Mover JavaScript, Extrair Função, Função Inline, Renomear, Alterar assinatura, puxar para cima, empurrar para baixo	
C	Mover, Extrair Função, Função Inline, Renomear, Alterar assinatura
Vai	Mover, Extrair Função, Função Inline, Renomear, Alterar assinatura

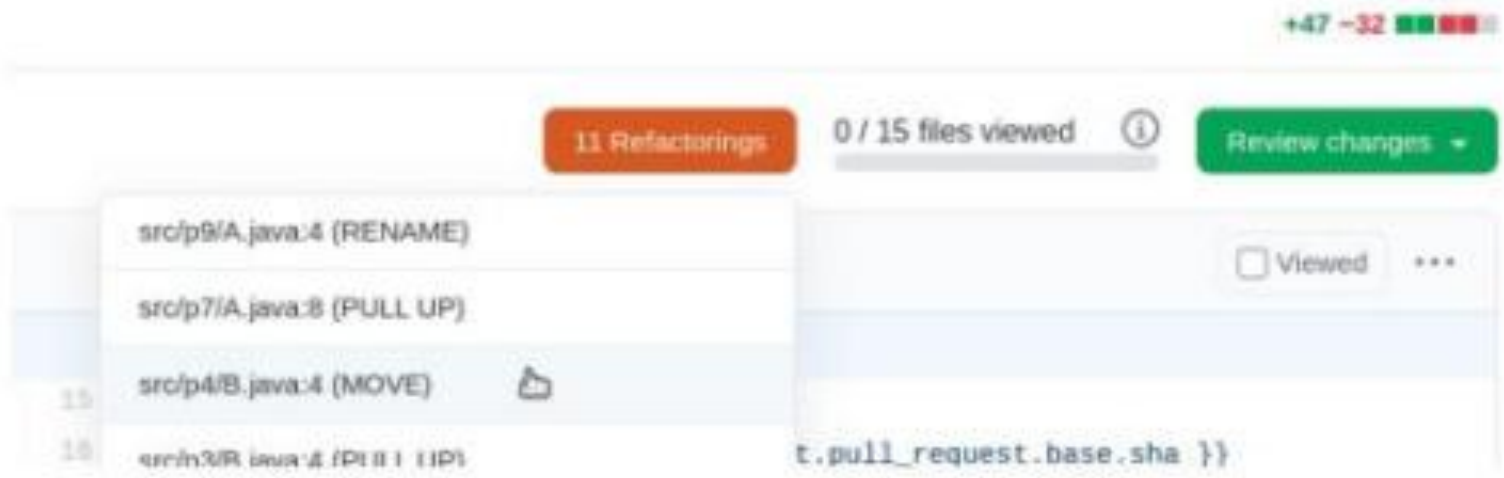


Fig. 5. RAID adiciona um botão na barra de ferramentas que fornece acesso fácil à lista de refatorações em uma solicitação pull

Arquitetura do RAID

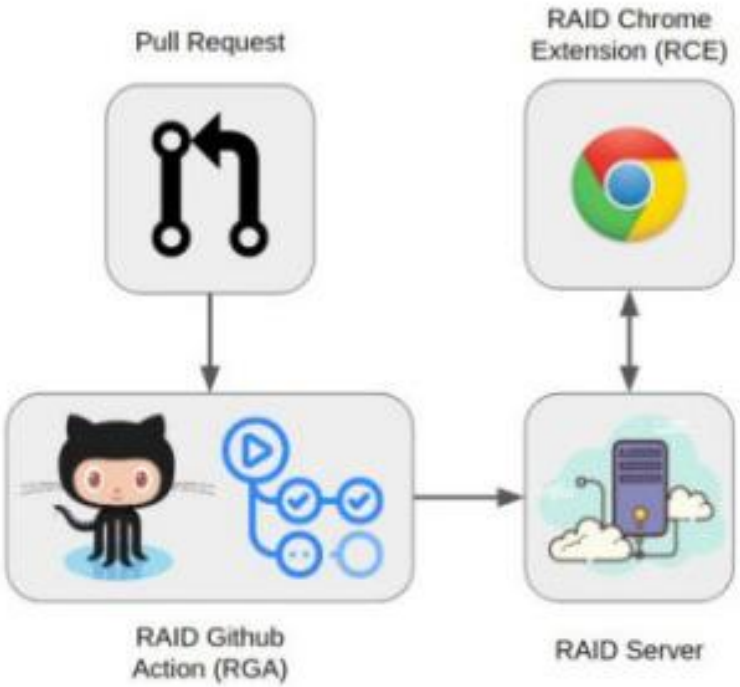


Fig. 6. Principais componentes do RAID e fluxo de execução

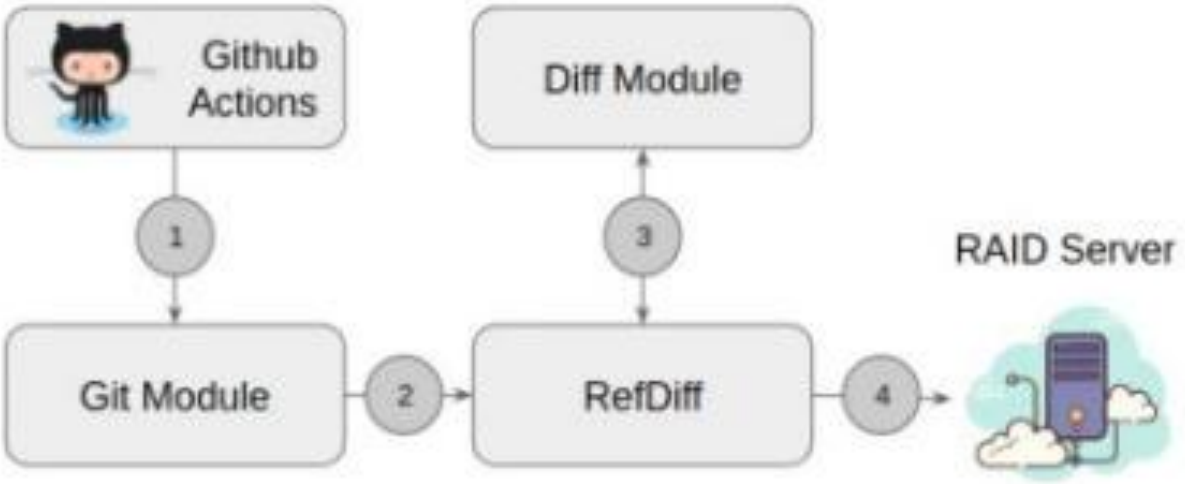


Fig. 7. Módulos RAID GitHub Action (RGA) e fluxo de trabalho

Experimento de campo

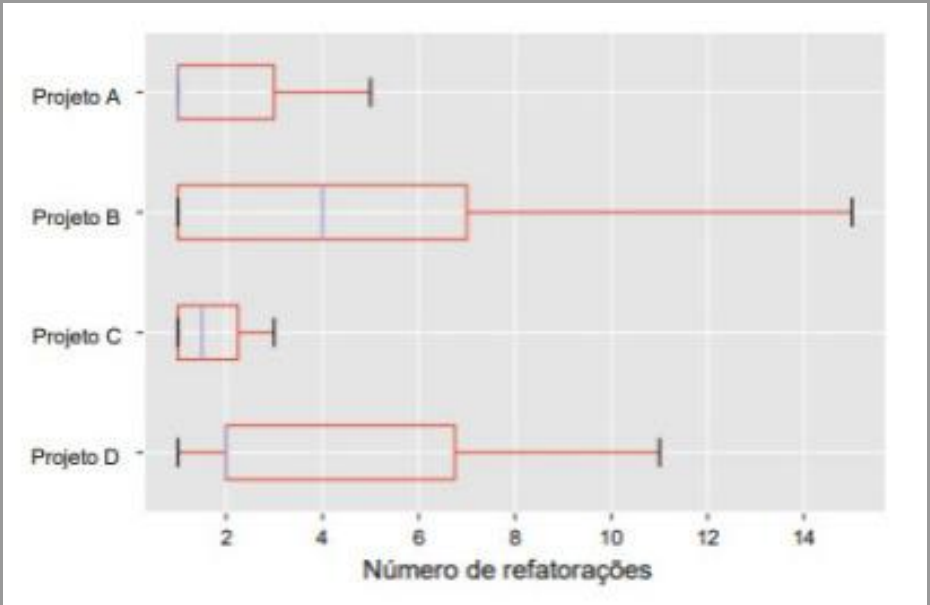
Metodologia

A metodologia usada para avaliar o RAID foi um experimento de campo, onde a ferramenta foi utilizada no fluxo de trabalho de desenvolvimento. Onde oito desenvolvedores profissionais a utilizam por três meses.

Antes de iniciar o experimento, os participantes receberam instruções e treinamento.
Em seguida, utilizaram o RAID por uma semana para se acostumarem e um dos autores forneceu suporte durante esse período.

Metodologia

Durante o experimento, foram criadas 325 pull requests, onde 84 incluíram refatoração



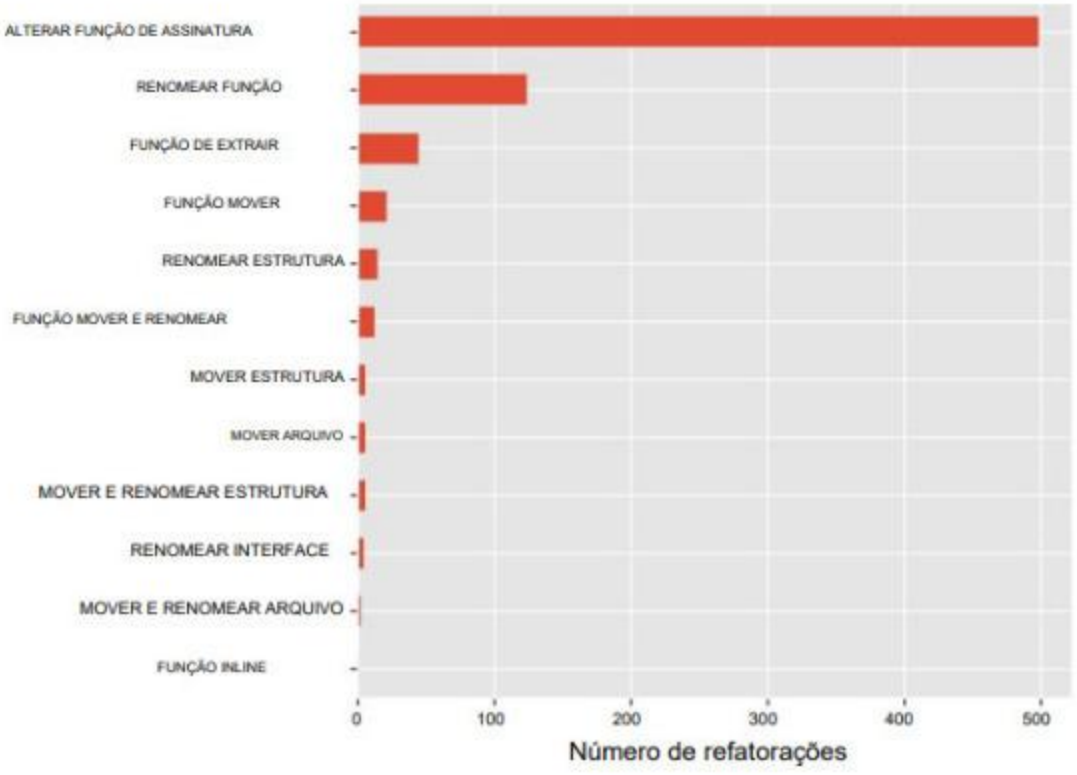


Fig. 9. Refatorações mais frequentes

Após o experimento, foi enviado perguntas aos participantes sobre o que achavam do RAID.
Foram feitas três perguntas:
→ Quais são os principais benefícios do RAID?
→ Quais são as dificuldades enfrentadas ao usar o RAID?
→ Algum comentário ou sugestão adicional?

Questões e Resultados de Pesquisa

RQ1: Qual é a sobrecarga de tempo de execução introduzida pelo RAID?

Questões e Resultados de Pesquisa

RQ2: Como os desenvolvedores usaram o RAID durante a revisão do código?

Questões e Resultados de Pesquisa

RQ3: Quanto esforço cognitivo é reduzido com RAID?

Questões e Resultados de Pesquisa

RQ4: Como os desenvolvedores perceberam o RAID?

Ameaças à validade

→ Validade Externa

→ Validade Interna

Conclusão

Neste artigo foi apresentado o RAID, uma ferramenta com reconhecimento de refatoração que instrumenta o GitHub diff com informações de refatoração.
Também foi realizado um experimento de campo com oito desenvolvedores profissionais durante três meses e concluímos que o RAID pode reduzir o esforço cognitivo necessário para revisar refatorações ao usar diferenças textuais.
Além disso, esse estudo relata uma redução no número de linhas necessárias para a revisão de tais operações.

Fim.

Obrigado!