AVALIAÇÃO DA QUALIDADE DE CÓDIGO EM *RUBY ON RAILS* COM *CODE CLIMATE*

SILVA, Guilherme Côrtes

VALENTE, Wander Antunes Gaspar

RESUMO

O trabalho proposto apresenta a ferramenta Code Climate, destinada à análise estática de código fonte para a linguagem Ruby on Rails. O artigo discorre sobre as características da ferramenta, metodologia que é utilizada para gerar as notas apresentadas e conceitos sobre análise estática. É apresentado um estudo de caso através da utilização da ferramenta em uma aplicação desenvolvida em Ruby on Rails. O estudo de caso consiste em obter métricas da ferramenta logo após o software ser desenvolvido, avaliar os resultados obtidos e submeter a aplicação a uma nova análise para verificar os novos resultados alcançados. Como conclusão é recomendado a utilização desta classe de ferramenta para melhoria do código fonte, além de permitir a simplificação e facilitar a manutenção.

Palavras-chave: Análise Estática. Code Climate. Qualidade de Código. Ruby on Rails. Refatoração.

1 INTRODUÇÃO

Escrever um bom código é de fundamental importância para manter a qualidade e facilitar a manutenção. Segundo Pressman (2011), um código mal escrito é um dos oponentes do mercado de software, sendo responsável por até 45% do tempo que um sistema fica inativo e custa bilhões de dólares com manutenção e redução de produtividade, não incluindo ainda a perda de clientes devido à insatisfação.

Durante a revisão do código, desenvolvedores tentam melhorar a qualidade do que foi escrito, corrigindo bugs ou fazendo com que o software

fique mais fácil de ser entendido para uma futura manutenção. Um método que pode auxiliar nesta tarefa são as ferramentas de análise estática [Panichella et al. 2015].

Análise estática é o processo de analisar estaticamente e de forma automatizada o código fonte de um programa para detectar possíveis erros, sem que o software precise ser executado [Prähofer et al. 2012].

Ferramentas de análise estática possuem a capacidade de identificar informações a respeito do software sem a necessidade de executá-lo. Isto torna as ferramentas apropriadas para a detecção de problemas no código, que podem auxiliar na inclusão de vulnerabilidades na aplicação. Devido a isto, tais ferramentas se tornam adequadas para o esforço de melhorar a qualidade do código [Ribeiro 2015].

No presente trabalho, uma aplicação desenvolvida em *Ruby on Rails* será submetida a uma ferramenta de análise estática de código denominada *Code Climate* para avaliar a qualidade da escrita. Após o primeiro resultado, o código será alterado para facilitar o entendimento e submetido novamente para análise.

2 METODOLOGIA

A pesquisa realizada neste trabalho pode ser considerada exploratória, devido ao estudo de caso e conceitos apresentados. Os dados foram coletados através do resultado da análise da ferramenta e analisados através das notas e gráficos gerados.

3 RESULTADOS E DISCUSSÃO

Code Climate foi desenvolvida em 2011, na cidade de Nova Iorque, e já analisou mais de 40 mil projetos *open source*. Todos os dias, estima-se que a ferramenta auxilia 50 mil desenvolvedores e analisa 700 bilhões de linhas de código, possuindo suporte a diversas linguagens tais como *Ruby*, *JavaScript*, PHP, *Python*, *Go*, *Ember.js*, *CoffeeScript*, CSS, *RubyMotion*, dentre outras [Helmkamp 2015].

A *Grade Point Average* (GPA) é uma métrica padrão utilizada nos Estados Unidos e em outros países para indicar um nível de performance acadêmica do usuário. É determinado um único número que resume a carreira acadêmica do estudante. Ainda que a GPA seja originalmente voltada para o ambiente acadêmico, pode ser usada em qualquer contexto. A ferramenta baseia-se neste conceito de GPA para atribuir notas aos projetos analisados. Em cada projeto é atribuído uma GPA com base na qualidade do código apresentado, sendo calculado a partir das notas de cada arquivo [Code Climate 2015].

De acordo com a ferramenta [Code Climate 2015], quanto mais próximo de quatro for a GPA, mais adequado estará o código segundo as métricas analisadas. A nota é gerada após analisar todos os arquivos *Ruby* do projeto, fazendo o cálculo de complexidade, duplicação de código, quantidade de linhas de código e número de vezes que o arquivo foi modificado. Também são verificadas questões de segurança, como por exemplo, *SQL Injection*, *Denial of Service*, *Cross Site Scripting*, entre outros [Akita 2014].

A aplicação foi desenvolvida com objetivo de controlar as tarefas de uma empresa e optou-se pela utilização de *Ruby on Rails,* pois é um *framework open source* que utiliza a arquitetura *Model-View-Controller* (MVC) e, segundo Lima (2014), possui como ênfase a velocidade e simplicidade no desenvolvimento de aplicações web.

O sistema utilizado está disponível no GitHub a partir do link https://github.com/guilhermecortes/HelpDesk e no *Code Climate* através da URL https://codeclimate.com/github/guilhermecortes/HelpDesk.

O software foi submetido a análise da ferramenta e obteve um GPA de 2.06. A ferramenta apontou cinco problemas com complexidade e dezoito de duplicação.

Após analisar os itens de duplicação, observou-se que todas as inconsistências apontadas foram para códigos similares, nenhuma ocorrência de código idêntico foi apresentada.

Em relação aos cinco itens de complexidade, três são referentes a métodos de criação de objetos, *Customers*, *Employees* e *Tickets*. Os outros dois itens pertencem à biblioteca que é utilizada para fazer o controle de usuários no sistema, denominada *Devise*.

Para solucionar os itens de complexidade e duplicação apresentados nos controladores, será utilizada a biblioteca *Responders*, que auxilia na redução de código das aplicações *Rails* a partir da versão 4.2. A biblioteca centraliza em um único local a forma dos controladores responderem a solicitação da aplicação e as mensagens de aviso, que são exibidas após a realização de qualquer ação resultante em alteração de valor no banco de dados.

Após a utilização da biblioteca *Responders* e, consequente, a alteração dos códigos identificados pela ferramenta que apontaram duplicação e complexidade, a aplicação foi submetida a uma nova análise e obteve um GPA de 3.92.

Após as alterações, todos os itens de duplicação e as três inconsistências de complexidade foram resolvidas. Permaneceram apenas as observações referentes aos métodos utilizados pela biblioteca *Devise*.

Para resolver as inconsistências restantes de complexidade da biblioteca Devise, será utilizada a extração de método, uma técnica de refatoração.

Extrair método é uma das técnicas de refatoração mais comuns que existem. Entende-se por métodos longos e complexos aqueles que possuem muitas responsabilidades e lógicas complexas, sendo necessário comentários para entender os objetivos que foram definidos. Para solucionar este problema deve-se então criar métodos menores, que utilizem fragmentos do anterior, e que o nome do método explique o objetivo [Fowler 1999].

Após a eliminação das complexidades restantes e consequente melhoria da nota do controlador *Registrations*, alcançou-se um GPA de 4.

4 CONCLUSÃO

Este trabalho apresentou uma análise técnica da ferramenta *Code Climate*, com objetivo de melhorar a qualidade do código desenvolvido em *Ruby on Rails*, eliminando itens de complexidade e duplicação que foram apontados.

Após a primeira análise realizada pela ferramenta, foram apontados problemas de código complexo e duplicado. Os itens identificados foram solucionados após a atribuição de responsabilidades a um único local na aplicação, pela utilização da biblioteca *Responders*, e após aplicar a técnica de refatoração, extração de método.

O resultado apresentado sugere que o uso de ferramentas como Code Climate podem ser recomendadas para uso em ambientes de desenvolvimento de software uma vez que oferecem recursos interessantes relacionados à melhoria do código fonte, devido principalmente à redução e simplificação, o que contribui para facilitar os trabalhos de manutenção.

REFERÊNCIAS

AKITA, Fábio. **Code Climate, Qualidade de Código e os Rubistas Sádicos**. 2014. Disponível em: <http://www.akitaonrails.com/2014/01/30/codeclimate-qualidade-de-codigo-e-os-rubistas-sadicos#.VdZ-lLSppHg>. Acesso em: 01 set. 2015.

DOCUMENTAÇÃO Code Climate. Disponível em: <http://docs.codeclimate.com>. Acesso em: 01 set. 2015.

FOWLER, Martin. **Refactoring**: Improving the Design of Existing Code. Addison-Wesley, 1 ed. 1999.

HELMKAMP, Bryan. **Introducing the Code Climate Platform**. Nova Iorque, 2015. Disponível em: <https://www.youtube.com/watch?v=lukVYdENeNY>. Acesso em 02 set. 2015.

LIMA, Johann Gomes Barros. **Uma aplicação de impacto social com aprendizagem de máquina**. UFRPE, Recife, 2014.

PANICHELLA, Sebastiano et al. **Would Static Analysis Tools Help Developers with Code Reviews?** Montréal, Canada, 2015.

PRÄHOFER, Herbert. et al. **Opportunities and Challenges of Static Code Analysis of IEC 61131-3 Programs**, Austria, 2012.

PRESSMAN, Roger. **Engenharia de Software**: Uma Abordagem Profissional. AMGH, 7. ed. Porto Alegre, 2011.

RIBEIRO, Athos Coimbra. **Análise Estática de Código-Fonte com Foco em Segurança**: Metodologia Para Avaliação de Ferramentas, Universidade de Brasília UnB, Brasília, DF, 2015.