

# ANÁLISE DE MÉTRICAS PARA AVALIAÇÃO DE QUALIDADE DE SOFTWARE COM O USO DE PADRÕES DE CÓDIGO

Valdiriano do Nascimento Pereira ([valdiriano@gmail.com](mailto:valdiriano@gmail.com))

Érika Höhn ([erika.hohn@gmail.com](mailto:erika.hohn@gmail.com))

## Resumo

Este trabalho visa a analisar métricas de código para gerenciamento da qualidade de software, no contexto de uso de padrões de código no desenvolvimento de software. Essa análise será feita comparando métricas de código utilizando dois frameworks diferentes, antes e depois de aplicar padrões de código, apresentando as diferenças entre as métricas extraídas, e como tais métricas podem demonstrar quantitativamente o nível de qualidade de software.

**Palavras-chave:** Qualidade de Software. Métricas. Padrões de Código.

## Introdução

De modo geral, a definição de qualidade é imprecisa e varia de acordo com o ponto de vista e a finalidade do objeto a ser avaliado. Segundo Kan (2002), em se tratando de software, a qualidade pode ser caracterizada pela falta de erros ou defeitos em um software, ou se o software atende os requisitos previamente definidos; porém, a qualidade de um software pode ser medida em várias dimensões diferentes.

Duas das preocupações que existem no gerenciamento de qualidade de software são estabelecer frameworks e padrões que elevem a qualidade do software, e definir as metas de qualidade a serem atingidas (SOMMERVILLE, 2011). Desta forma, juntamente com os processos de desenvolvimento, devem ser definidos e escolhidos os padrões que serão utilizados. Entre esses padrões, estão alguns já existentes e consolidados, regulamentados pela ISO ou IEEE, por exemplo, ou padrões próprios internos da equipe.

Em muitos casos, garantir qualidade de software significa somente definir tais processos, procedimentos e padrões; em outros, a garantia da qualidade inclui também gestão de configuração, verificação e validação, mesmo após a entrega do produto (SOMMERVILLE, 2011). Assim sendo, o código fonte do software deve seguir os padrões pré-definidos e escolhidos pela equipe, de forma que tenha características que auxiliem na compreensão e manutenção do software pela equipe (PRESSMAN, 2011).

Sendo assim, o objetivo deste trabalho é analisar métricas de código e apresentar as diferenças nos resultados de métricas de software seguindo padrões e softwares sem tais padrões. Para realizar essa análise, foram extraídas métricas de código em um projeto utilizando dois frameworks diferentes, sem utilizar padrões de código, e então o código foi refatorado para se adequar aos padrões, sendo medido novamente.

## 1. Referencial teórico

### 1.1. Qualidade de Software

De acordo com Pressman (2011), qualidade de software pode ser definida como “uma gestão de qualidade efetiva aplicada de modo a criar um produto útil que forneça valor

mensurável para aqueles que o produzem e para aqueles que o utilizam”. Essa definição pode ainda ser mais detalhada e alterada de acordo com várias características a serem medidas em um produto de software.

## 1.2. Métricas

## 1.3. Padrões de código

### 1.3.1. PSR

PSRs (PHP Standards Recommendations) são guias de estilo para PHP criados pelo PHP-FIG (Framework Interoperability Group), um grupo cujo ideal é manter a comunicação entre a comunidade de desenvolvedores PHP, de forma que todos possam conversar e encontrar pontos em comum. O FIG é composto por grandes representantes da comunidade PHP, responsáveis por projetos bastante conhecidos e utilizados, como o CakePHP, Doctrine, Drupal, entre outros.

Cada recomendação (PSR) aborda um assunto, desde o básico, até assuntos mais avançados e específicos, como cache, links, autoloading etc. Existem atualmente 8 PSRs aprovadas, enquanto outras ainda estão em fase de rascunho ou revisão.

As PSRs que abordam recomendações referentes a código são as PSR-0, PSR-1, PSR-2 e PSR-4, e já são utilizadas por grandes projetos utilizando PHP, como Drupal, Zend, Symfony, Laravel, CakePHP, phpBB, AWS SDK, FuelPHP, Lithium etc (PHP, 2017).

## 2. Metodologia

Para fazer a análise de métricas de código e poder extrair dados iniciais, serão utilizadas duas versões de um mesmo projeto em PHP, uma delas utilizando o framework CakePHP, versão 2.x, e a outra utilizando o framework CodeIgniter, versão 3.x. As duas versões do projeto serão equivalentes na finalidade e nas funcionalidades, para que a comparação dos resultados seja feita de forma justa.

Esses dois projetos a princípio não estarão seguindo os padrões PSR, e algumas métricas de código serão extraídas dos projetos neste estado. Então, os dois projetos serão ajustados para se adequarem ao PSR, e, após isso, os dois projetos serão avaliados novamente e métricas pós-ajuste serão extraídas. Desta forma, os projetos podem ser avaliados antes, durante e depois dos ajustes necessários para estarem seguindo os padrões.

O CakePHP, por si só, já exige que alguns padrões próprios do framework sejam utilizados para que haja o correto funcionamento do código, mas essa padronização se limita à nomenclatura de métodos e classes. Ademais, a escrita do código é livre, de acordo com o desenvolvedor.

Por outro lado, o CodeIgniter não necessita que haja uma padronização de nenhuma forma, desde que a estrutura de pastas e a hierarquia de classes sejam respeitadas.

Esses dois itens foram decisivos para a escolha destes dois projetos e frameworks, de forma que a análise a que este trabalho se propõe consiga analisar pelo menos dois cenários diferentes.

Várias ferramentas de análise de projetos e levantamento de métricas foram encontradas, entre elas, a Kalibro, Analizo, e outras, porém somente três serão utilizadas para efeito de análise: phpdefend, phpmetrics e phpcs. As duas foram selecionadas por tratarem

especificamente de projetos em PHP, pela quantidade de métricas abordadas, forma de apresentação dos resultados e por estarem atualizadas.

## 2.1. Ambiente de testes

Para a realização dos testes propostos, foram utilizadas as seguintes ferramentas:

- Servidor Apache v x.x;
- PHP v 7.x;
- Sistema operacional Ubuntu 16.x;
- Editor de texto Sublime Text 3;
- IDE PHP Storm.

## 2.2. Estudo de caso

O projeto utilizado será uma loja, contendo já alguns produtos com respectivos dados para compra; cada produto pertence a uma categoria.

É permitido ao usuário adicionar e remover produtos do carrinho, e prosseguir para o checkout.

Abaixo segue diagrama de classes da loja.

Esta loja é baseada no projeto, em CakePHP, encontrado no repositório de arquivos GitHub, no link: <https://github.com/andaskende/cakephp-shopping-cart>.

## 2.3. Métricas extraídas

Para prosseguir com a análise das métricas, foram extraídas métricas de código utilizando \*. Segue abaixo a relação de métricas levantadas e analisadas:

## 3. Resultados obtidos

## 4. Conclusão

## 5. Referências

CAKE SOFTWARE FOUNDATION INC. **CakePHP:** Cookbook. Disponível em: <<https://book.cakephp.org/2.0/pt/getting-started/cakephp-conventions.html>>. Acesso em: 31 mar. 2017.

KAN, Stephen H.. Metrics and Models in Software Quality Engineering. 2. ed. Boston: Addison Wesley, 2002.

PHP-FIG. Disponível em: <[www.php-fig.org/](http://www.php-fig.org/)>. Acesso em: 30 mar. 2017.

PHP: The Right Way. The Right Way. Disponível em: <[http://www.phptherightway.com/#code\\_style\\_guide](http://www.phptherightway.com/#code_style_guide)>. Acesso em: 16 abr. 2017.

PRESSMAN, Roger S.. **Engenharia de Software:** Uma abordagem profissional. 7. ed. Porto Alegre: Amgh, 2011.

SOMMERVILLE, Ian. **Engenharia de Software.** 9. ed. São Paulo: Pearson Prentice Hall, 2011.