Extração da Métrica WMC a partir de Código Java

Luana V. Martinez, Maurício S. R. Arrieira, Carlos M. Betemps

Engenharia de Computação – Universidade Federal do Pampa (UNIPAMPA)

Campus Bagé – RS – Brasil

{luanavmartinez, mauriciorealan, carlos.betemps}@gmail.com

Abstract. *Software metrics are measures of software system properties that allow a quantitative evaluation of a software element under analysis. Aiming to describe the software metrics concepts, this work addresses the related techniques, mainly the object oriented metrics. Also, it proposes an approach for the WMC metric extraction, and presents a software prototype that performs such extraction.*

Resumo. *As métricas de software consistem em medidas das propriedades de um sistema de software, permitindo uma avaliação quantitativa de um elemento de software sob análise. Buscando descrever pontualmente os conceitos relacionados às métricas de software, esta pesquisa visa abordar as técnicas referentes ao tema, com ênfase nas Métricas Orientadas a Objetos, e apresenta uma abordagem para a extração da métrica WMC e um protótipo de software com capacidade de realizar tal extração.*

1. Introdução

Atualmente o software é considerado um dos maiores influenciadores do orçamento das organizações. A grande maioria delas reconhece a importância de controlar os gastos com software, analisando o desempenho dos resultados obtidos com o desenvolvimento e a manutenção dos mesmos. Desta maneira, é necessário fazer uso de medidas e de modelos apresentados na Engenharia de Software [Cordeiro 2008].

Para obter-se um software de qualidade é necessário realizar medidas nesse software, também conhecidas como métricas de software. Uma métrica é qualquer tipo de medição que se refira a um sistema de software, processo, ou documentação relacionada [Sommerville 2003]. As métricas de software são expressas de forma quantitativa, ou seja, em números, sem elas os dados obtidos seriam apenas dados subjetivos. Segundo Guarizzo [2008], com a medição feita é possível estimar custos e prazos para o desenvolvimento do projeto e entrega do produto final.

O crescimento constante da demanda de desenvolvimento de software motivou um estudo e análise das métricas de software. Assim, modelos e ferramentas capazes de realizar a medição das estruturas de um software se tornam cada vez mais imprescindíveis. Dessa maneira, o objetivo fundamental deste trabalho é apresentar uma técnica capaz de extrair a métrica de software WMC (Weighted Methods per Class – Métodos Ponderados por Classe). Sendo assim, será apresentada uma análise introdutória sobre as métricas de software, com foco nas métricas orientadas a objetos, e a apresentação de uma abordagem para a extração da métrica WMC de códigos Java.

O restante deste trabalho está organizado da seguinte forma: na seção 2 o referencial teórico do trabalho é apresentado. Em seguida, a seção 3 apresenta a metodologia de desenvolvimento deste trabalho. Na seção 4 é mostrada a abordagem proposta para a extração da métrica WMC. A seção 5 apresenta o protótipo de software desenvolvido. Por fim, as conclusões e trabalhos futuros são apresentados na seção 6.

2. Fundamentação Teórica

Para embasar este estudo faz-se necessário um levantamento teórico de aspectos relevantes ao tema, os quais são apresentados nesta seção, iniciando com as Métricas de Software, seguida de Métricas de Software Orientadas a Objetos e as Métricas CK [Chidamber e Kemerer 1994], além dos conceitos relacionados aos métodos para extração de métricas.

2.1. Métricas de Software

De acordo com Balzer et al. [2005], métricas de software são medidas quantitativas do grau em que um sistema de software, um componente, ou processo possuem um determinado atributo. Elas permitem a identificação de áreas problemáticas, a ilustração de tendências e, assim, ajudam a melhorar a qualidade dos produtos de software, bem como ajudam a aumentar a eficiência do processo de desenvolvimento. No contexto de compreensão de programa e análise de software muitas métricas de software têm sido estudadas. Exemplos de métricas de software podem ser o número de linhas de código e o número de atributos públicos em uma classe.

Guarizzo [2008] apresenta algumas vantagens e desvantagens relacionadas ao uso de métricas de software, conforme apresenta a Figura 1. Conforme o autor, a desvantagem no uso de métricas está relacionada à confiabilidade dos resultados obtidos na extração das mesmas.

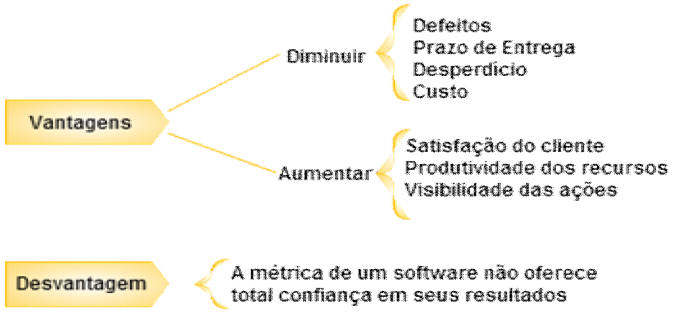

Figura 11. Vantagens x Desvantagens no Uso de Métricas de Software

2.2. Métricas Orientadas a Objetos

Cada vez mais as métricas orientadas a objetos têm sido usadas para avaliar a qualidade do software, assim indica Harrison *et al.* [1998]. Para se tornar válida, uma métrica precisa provar que mede exatamente o que propôs medir.

O conjunto de métricas CK [Chidamber e Kemerer 1994] foca nas propriedades da estrutura base de projetos OO - a classe - e buscam medir características de tamanho, acoplamento, coesão e herança nas classes de um projeto de software.

2.3. Métricas de Chidamber e Kemerer

As métricas CK são métricas orientadas a classe [Chidamber e Kemerer 1994]. São compostas por métricas que permitem a análise quantitativa dos artefatos de software construídos utilizando o paradigma da orientação a objetos. O objetivo dessas métricas é salientar as classes que possivelmente contém maior número de defeitos, com o propósito de direcionar os esforços de teste.

Oliveira [2012] apresenta que Chidamber e Kemerer propuseram um dos conjuntos mais amplamente conhecidos de métricas de software orientado a objetos. Neste conjunto estão as métricas:

- Métodos ponderados por classe (*Weighted Methods per Class* – WMC);
- Profundidade da Árvore de Herança (*Depth of the Inheritance Tree* – DIT);

- Número de filhas (os) (N*umber Of Children* – NOC);
- Acoplamento entre Classes de Objeto (*Coupling Between Objects* – CBO); e
- Falta de Coesão em Métodos (*Lack of Cohesion in Methods* – LCOM).

2.4. Métrica WMC (*Weighted Methods per Class*)

A métrica WMC – Métodos Ponderados por Classe – é uma preditora de quanto tempo e esforço são necessários para desenvolver e manter uma classe. Esse método atribui pesos aos métodos de uma classe e, dessa maneira, é medida a complexidade individual de cada classe [Sommerville 2003]. Se todos os métodos de uma classe apresentarem a mesma complexidade (1, por exemplo), então WMC será somente o número de métodos definidos em cada classe [Pfleeger 2004]. Operações herdadas de uma superclasse são desconsideradas.

É possível medir os métodos de acordo com o número de linhas de código do método, complexidade ciclomática, número de parâmetros, entre outros. Assim, o número de métodos e a sua complexidade relacionada indicam o tempo e o esforço requerido para a programação e a manutenção da classe.

2.5. Métodos para Extração de Métricas

A grande quantidade de métricas, coletas manuais e poucos recursos de visualização são fatores que acabam por desmotivar o uso destas para o monitoramento do código. Além disso, a compreensão do significado de valores obtidos por meio de métricas não é uma tarefa trivial, demandando um grande esforço de interpretação necessário para a tomada de decisão efetiva sobre o projeto de software.

Assim, destaca-se a importância de ferramentas que auxiliem o processo de medição, compreensão e visualização do software. Atualmente existem algumas ferramentas que automatizam a extração de métricas do código fonte, com objetivo de coletar as informações sobre o produto a partir da análise estática do código. Estas são definidas como ferramentas *Extratoras* [Del Esposte 2014].

3. Metodologia

Como o objetivo geral deste trabalho é realizar um estudo das métricas de software, com foco na métrica WMC, foi adotada uma sequência de passos metodológicos para que, com uma busca e análise do referencial teórico e estudo especifico sobre métricas de software, fosse possível desenvolver uma abordagem e respectiva aplicação que fossem capazes de extrair a métrica WMC de código fonte escrito em Java. Na Figura 2 está demonstrada a sequência metodológica adotada para este trabalho.

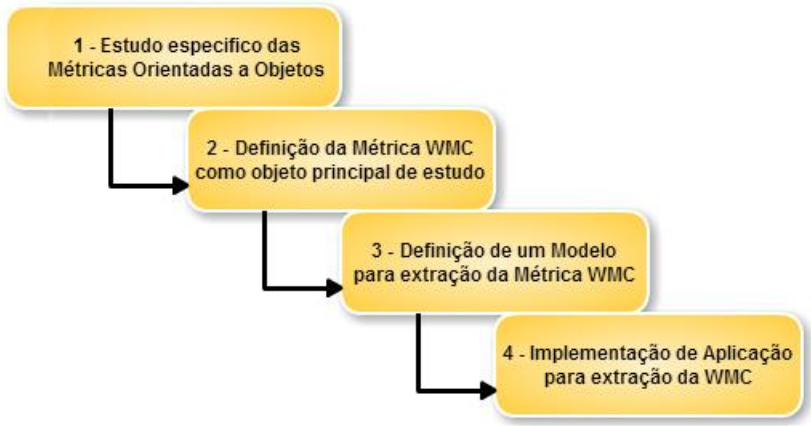

Figura 2. Sequência de Passos da Metodológica Utilizada no Trabalho

Em um primeiro momento, foi realizado o levantamento de referências existentes sobre o tema abordado utilizando mecanismos de busca na internet, bibliotecas digitais (como a *IEEExplore*[10]) e livros da área de engenharia de software. Posteriormente, foi feita a análise deste referencial. Em seguida foram estudadas as métricas de software existentes. O próximo passo foi separar dentre essas métricas encontradas as métricas orientadas a objetos e, consequentemente, a métrica Métodos Ponderadas por Classe (WMC). Tendo conhecimento sobre a métrica WMC, a mesma foi escolhida como objeto principal de estudo e experimentação neste trabalho. Em momento posterior foi desenvolvida uma abordagem para extração de tal métrica. Por fim, foi realizado o desenvolvimento de um programa em Java, utilizando o editor de código fonte *Notepad++*, com funcionalidade de extração da métrica WMC, conforme definida na abordagem.

4. Abordagem de Extração da Métrica WMC

A linguagem de programação Java foi usada como referência neste projeto, pois seu modelo de constituição classes-métodos possibilita uma avaliação eficaz da métrica WMC. A abordagem para extração da métrica WMC apresentada neste trabalho tem por princípio inicial realizar uma análise sobre o código fonte (escrito em Java).

Assim, para a avaliação da métrica, o primeiro passo da abordagem é identificar cada palavra do texto do código fonte, com intuito de ter uma referência para os símbolos e possíveis palavras reservadas da respectiva linguagem de programação.

Em um segundo momento se realiza uma comparação entre todas as palavras do texto do código fonte com palavras reservadas que podem integrar a assinatura de um método. No caso da linguagem Java a comparação é feita com palavras reservadas da linguagem, seguindo a estrutura de um método Java. A Figura 3 apresenta o modelo de estruturação de um método, baseado no descrito por Gosling e seus colegas no livro *Java Language Specification* [Gosling et al. 2014], e alguns exemplos de assinatura de métodos. Neste modelo, antes do nome do método, a assinatura pode ser composta por: modificadores seguidos de um tipo de retorno ou somente por um tipo de retorno.

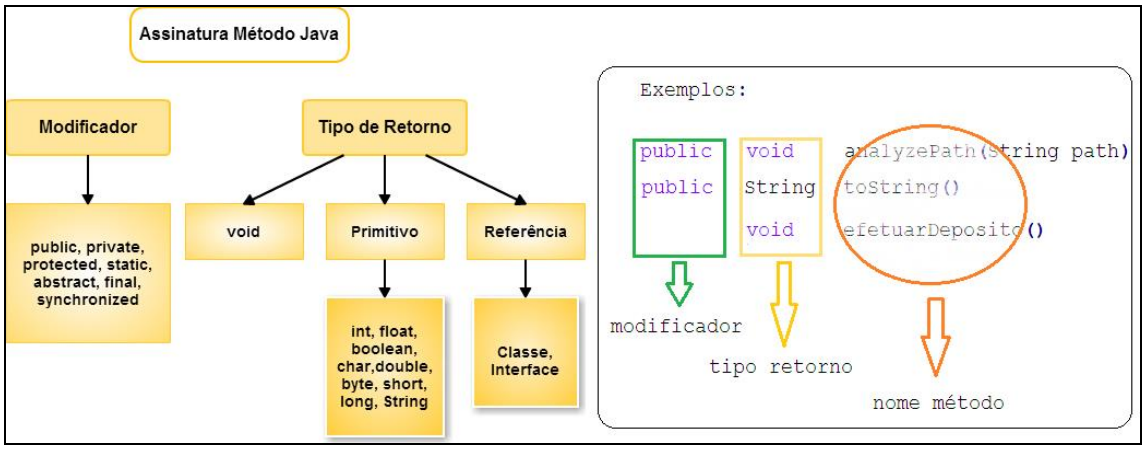

Figura 3. Modelo de Assinatura de um Método Java.

Quando uma comparação retorna um resultado positivo passa-se para mais uma comparação, isto porque alguns métodos apresentam mais de uma destas palavras reservadas em sua assinatura (caso dos modificadores). Para finalizar o passo de comparação e definir que um novo método foi encontrado, busca-se a palavra que está

[10] *http://ieeexplore.ieee.org/*

entre a palavra referente ao tipo de retorno e o símbolo de parênteses - este é, então, o nome do método.

Uma vez que a maneira de encontrar os métodos da classe já está definida, o próximo passo para a extração da métrica está em analisar a complexidade dos métodos. Neste trabalho a complexidade dos métodos de uma classe será ponderada pelo número de linhas do método.

Depois que todos os métodos da classe já possuem um número de linhas conhecido é realizada a atribuição do peso para cada método. A ponderação para cada método consiste em uma progressão que acrescenta uma unidade ao peso do método a cada cinco linhas, isto é, se um método possuir entre uma e cinco linhas ele vai possuir o peso "1", se ele possuir entre seis e dez linhas ele vai receber peso "2", se ele tiver entre onze e quinze linhas peso "3" e assim sucessivamente.

Por fim, depois que todos os métodos da classe já foram encontrados e ponderados ocorre, enfim, o cálculo da métrica WMC, onde o valor da mesma será o resultado do somatório dos pesos de cada método dividido pelo número de métodos que a classe possui.

A Figura 4 demonstra o processo para extração da métrica. Cada um dos passos da técnica está exposto e também é apresentado que, após o passo "3", já foram identificados o número de métodos da classe e o nome de cada método, fatores esses importantes no estágio 6 do processo.

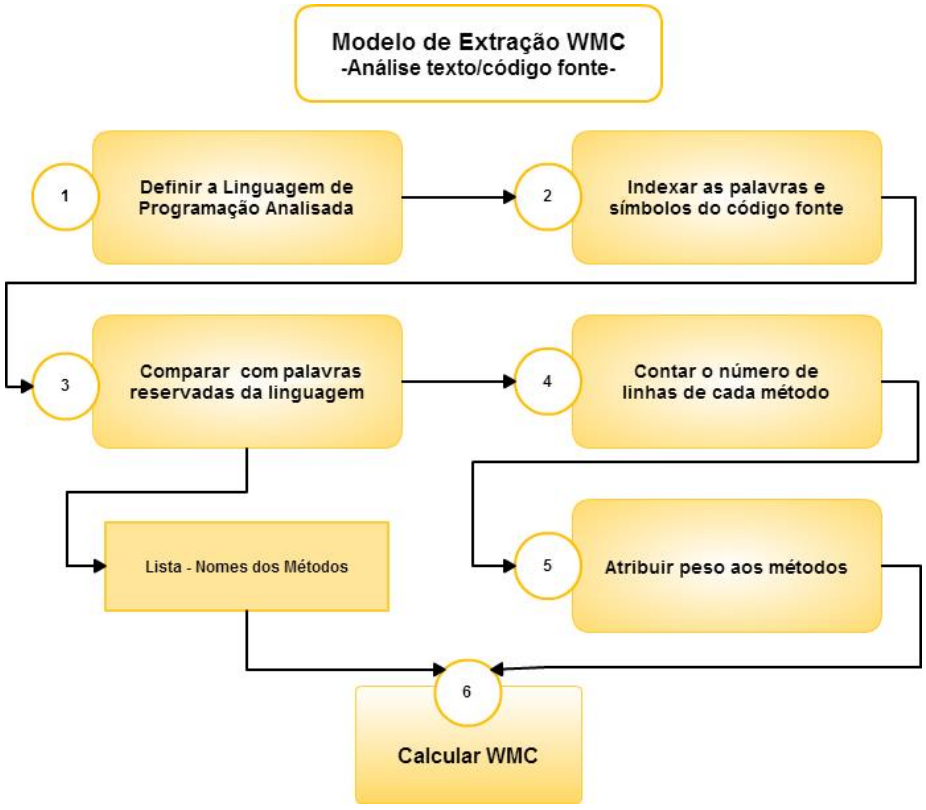

Figura 4. Modelo de Extração para métrica WMC.

5. Protótipo de Software Implementado

Para auxiliar na análise sobre o tema, foi construído um software utilizando a linguagem de programação Java, que extrai a métrica de software WMC, seguindo como premissa o modelo de extração estabelecido na Seção 4. O código desta implementação está dividido em classes, são ao todo nove classes que constituem a implementação. As

ferramentas utilizadas para realizar a implementação do programa foram o Notepad++ e NetBeans IDE 7.4, além do JDK. A implementação possui quatro classes que realizam as operações de processamento, seleção e carregamento de arquivos e também as classes que montam as interfaces gráficas.

Inicialmente, o software realiza a leitura dos códigos Java indicados pelo usuário. Esses arquivos são analisados e todas as suas linhas são decompostas em palavras. Uma vez que todas as palavras do arquivo possuem uma referencia única, o software começa uma análise em busca das referências que contenham o nome da classe e dos métodos de cada um dos arquivos lidos. Em um segundo momento, já com o conhecimento das classes e respectivos métodos, o protótipo realiza a contagem do número de linhas que cada método possui. Para cada método é atribuído um peso, conforme o critério estabelecido pela abordagem - número de linhas que o mesmo possui e progressão dos pesos a cada cinco linhas.

Após a análise dos códigos Java, uma estrutura em forma de árvore para análise simplificada é montada pelo software. Esta árvore apresenta uma constituição na qual cada uma das classes representa um ramo principal. Em cada um desses ramos existem os nós adjacentes correspondentes a cada um dos métodos da respectiva classe, e dentro de cada método seus respectivos números de linhas e peso ponderado. Por fim, o software realiza o cálculo da métrica WMC, conforme apresentado na seção 4.

As Figuras 5 e 6 demonstram o funcionamento do software implementado, ressaltando a estrutura de árvore utilizada (Figura 5) e os dados referentes à extração da métrica (Figura 6).

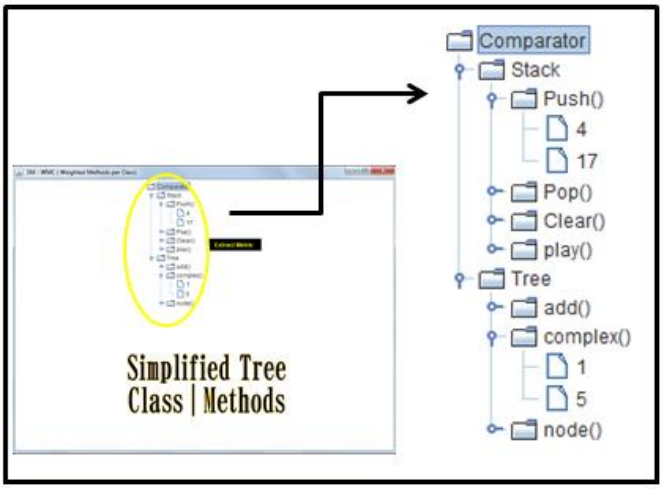

Figura 5. Estrutura em árvore utilizada na aplicação.

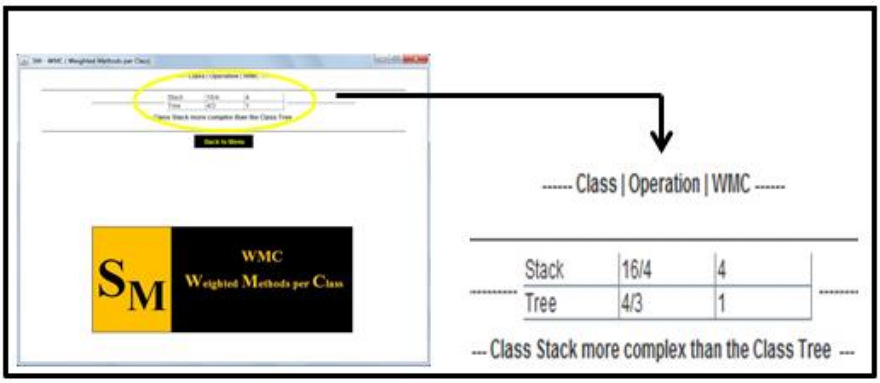

Figura 6. Exemplo de dados extraídos pela aplicação.

6. Conclusões e Trabalhos Futuros

Ao finalizar esta pesquisa, percebe-se que a abordagem para a extração da métrica WMC formulada neste trabalho pode servir como base para a implementação de aplicações que executem a extração da métrica WMC, como o protótipo implementado neste trabalho. O estudo de diferentes métricas de software indica a importância de tal conhecimento para uma melhor avaliação técnica do significado dos valores de métricas extraídos a partir de código fonte, como os valores da métrica WMC, que indicam a complexidade de uma determinada classe. O uso do número de linhas de um método, especificamente a atribuição do peso de um método com base no número de linhas, permite que um valor específico seja atribuído por quem está realizando as medidas, podendo ser ajustado e customizado conforme a iniciativa de desenvolvimento.

Por fim, a generalização do modelo apresentado de forma que possa ser utilizado em outras linguagens de programação, bastando indicar aspectos da sintaxe da linguagem, e, também, a definição de abordagens que tratem da extração de outras métricas, como as demais métricas CK, serão objetos de estudo em trabalhos futuros.

Referências

Balzer, Michael, Oliver Deussen, and Claus Lewerentz. "Voronoi treemaps for the visualization of software metrics." Proceedings of the 2005 ACM symposium on Software visualization. ACM, 2005.

Chidamber, A. R.; Kemerer, C. F. 1994, A Metrics Suite for Object-Oriendes Design, IEEE Trans. Software Engineering, vol SE-20, n 6, June, pp. 476-493.

Cordeiro-GPS, Marco Aurélio. “Métricas de Software”, 2008. Bate Byte. URL: Disponível em: <http://www.batebyte.pr.gov.br/modules/conteudo/conteudo.php?conteudo=88>. Acesso em: 23/10/2015.

Del Esposte, Arthur de Moura. Tomada de decisões orientadas a métricas de software: observações de métricas de produto e vulnerabilidades de software via DW e Plataforma de monitoramente de código-fonte. Monografia. Universidade de Brasília, 2014. Disponível em: <https://fga.unb.br/articles/0000/5535/TCC1_Arthur_e_Carlos.pdf>. Acesso em: 23/10/2015.

Gosling, James, et al. The Java Language Specification. Pearson Education, 2014.

Guarizzo, Karina. Métricas de Software. Monografia de Graduação, Faculdade de Jaguariúna, Jaguariúna, SP, 2008.

Harrison, R., Counsell , SJ, e Nithi , RV, " An evaluation of the MOOD set of object-oriented software metrics," IEEE Transactions on Software Engenharia , vol. 24, pp. 491-496, Junho de 1998.

Pfleeger, Shari Lawrence. Engenharia de software: teoria e prática. 2ª Edição, Pearson - Prentice Hall, 2004.

Sommerville, Ian. Engenharia de Software. Vol 6. São Paulo: Addison Wesley, 2003.

Oliveira, J. F. de. Métricas para Avaliação do Grau de Quantificação de Sistemas Orientados por Aspectos. Dissertação. Pontífica Universidade Católica de Minas Gerais – Programa de Pós-Graduação em Informática. Belo Horizonte, 2010