RAFTool - Ferramenta de Filtragem de Métodos, Classes e Pacotes com Medições Incomuns de Métricas de Software

Tarcísio G. S. Filó[1], Mariza A. S. Bigonha[1], Kecia Aline Marques Ferreira[2]

[1]Departamento de Ciência da Computação - Universidade Federal de Minas Gerais (UFMG) - Belo Horizonte - MG - Brasil

[2]Departamento de Computação - Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG) - Belo Horizonte - MG - Brasil

{tfilo, mariza}@dcc.ufmg.br, kecia@decom.cefet.mg.br

Resumo. *Este artigo apresenta a ferramenta RAFTool (Risk Artifacts Filter Tool), cujo propósito é realizar a filtragem de métodos, classes e pacotes que possuem medições anômalas de métricas de softwares orientados por objetos. Essa ferramenta apoia a realização de análise de medições coletadas sobre as estruturas internas do código-fonte em softwares orientados por objeto, podendo ser utilizada em Processos de Medição - MED do nível F (Gerenciado) do MR-MPS-SW. Uma das consequências da sua efetiva aplicação para as organizações é transformar dados de métricas de código-fonte em informações úteis para apoiar decisões relacionadas ao controle e avaliação da qualidade interna de software dentro da organização. RAFTool foi desenvolvida em Java e é uma ferramenta open source.*

Palavras-chave: *métricas, valores referência, qualide de software, medição de software, análise, apoio à decisão.*

1. Introdução

Este artigo descreve a ferramenta *RAFTool* (*Risk Artifacts Filter Tool*), cujo propósito é realizar a filtragem de métodos, classes e pacotes que possuam medições anômalas de métricas de softwares orientados por objetos, no contexto do processo de medição de software. Medições anômalas são aquelas que se afastam significativamente do que é comum, podendo indicar problemas de qualidade no artefato medido [Sommervile 2010]. Um processo de medição que pode fazer parte de um processo de controle de qualidade de software é exibido na Figura 1. Um estágio chave nesse processo é a identificação de medições anômalas, que é feita por meio da comparação das medidas obtidas com um histórico de medições de projetos anteriores, com o objetivo de identificar valores que estejam fora da normalidade. Os valores referência propostos por Filó *et al.* (2014) são capazes de indicar o que são esses valores "fora dos limites definidos como desejáveis" para as métricas. Isso facilita a aplicação desse processo de controle, afinal, não é prática comum das organizações a manutenção de um banco de dados de medições de projetos anteriores, como idealiza Sommerville (2010).

Além da definição de valores referência ser de suma importância para a efetiva aplicação das métricas na indústria de software, o fornecimento de ferramentas apropriadas de medição e análise quantitativa de software é fundamental, pois de acordo com Sommerville (2010), o processo de

medição deve ser simples e fácil. Para a medição de softwares orientados por objetos, existem ferramentas que têm sido utilizadas em pesquisas relacionadas a métricas de softwares, tais como *Eclipse Metrics Plugin*[1] e *Google CodePro AnalytiX*[2]. Essas ferramentas realizam a medição de métricas de software, porém, não fornecem meios para realizar uma análise quantitativa dos resultados da medição. Nesse contexto, foi desenvolvida a ferramenta *RAFTool*, que suporta, baseada em uma entrada de medições de software em formato XML, a identificação de métodos, classes e pacotes que apresentam medições anômalas de métricas de softwares orientados por objetos. *RAFTool* é um complemento do trabalho de Filó *et al.* (2014), que propuseram e aplicaram um método para identificar valores referência para um conjunto de métricas de softwares orientados por objetos. O restante desse artigo está organizado da seguinte forma: Seção 2 apresenta os valores referência para métricas de software utilizados por *RAFTool*. Seção 3 descreve o apoio de *RAFTool* às atividades do modelo MR-MPS-SW. Seção 4 apresenta os requisitos e funcionalidades da ferramenta. Seção 5 mostra a arquitetura utilizada no desenvolvimento de *RAFTool*. Seção 6 traz a conclusão, analisando as aplicações da ferramenta e extensões futuras.

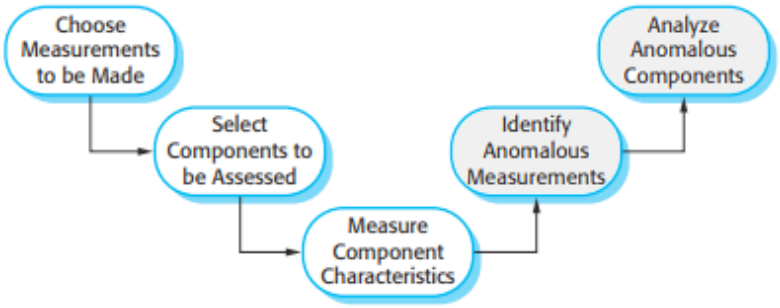

Figure 1. Processo de Medição. [Sommervile 2010].

2. Valores Referência

Filó *et al.* (2014) propõem valores referência para um conjunto de métricas de softwares orientados por objetos por meio de três faixas de valores: *Bom/Frequente*, *Regular/Ocasional* e *Ruim/Raro*. A faixa *Bom/Frequente* corresponde a valores com alta frequência, caracterizando os valores mais comuns da métrica, na prática. A faixa *Ruim/Raro* corresponde a valores com baixa frequência, e a faixa *Regular/Ocasional* é intermediária, correspondendo a valores que não são nem muito frequentes nem raros. Por exemplo, para a métrica *Número de Métodos* (NOM) têm-se os seguintes valores: *Bom/Frequente* (NOM $\leq$ 6), *Regular/Ocasional* (6 < NOM $\leq$ 14) e *Ruim/Raro* (NOM > 14). Os valores referência não expressam necessariamente as melhores práticas da Engenharia de Software, e sim um padrão seguido pela maioria dos softwares. Com a utilização dos valores referência desse trabalho de pesquisa, *RAFTool* irá informar ao usuário quais são os artefatos em risco baseado em aspectos quantitativos do código-fonte por meio da combinação dos valores referência de uma ou mais métricas.

1 http://metrics2.sourceforge.net/
2 https://developers.google.com/java-dev-tools/codepro/doc/

3. O Processo de Medição no MPS.BR

O propósito do Processo de Medição - MED do Nível F (Gerenciado) do MR-MPS-SW é coletar, armazenar, analisar e relatar os dados relativos aos produtos desenvolvidos e aos processos implementados na organização e em seus projetos, de forma a apoiar os objetivos organizacionais [Softex 2012]. *RAFTool* é uma ferramenta que apoia a análise dos dados coletados dentro do Processo de Medição, permitindo gerenciar aspectos essenciais da qualidade interna do código em softwares orientados por objetos, identificando métodos, classes e pacotes que mostrem alguma ameaça à qualidade interna do software. Dessa forma, *RAFTool* proporciona uma real conexão entre a coleta de dados e a sua utilização para apoiar decisões dentro da organização, permitindo que o MED implementado tenha impactos positivos nas organizações.

4. Requisitos

Nesta seção são especificados os requisitos funcionais de *RAFTool*.

4.1. Instanciação do Sistema Analisado

RAFTool permite a instanciação de um sistema a ser analisado a partir da digitação de um nome. Posteriormente, ela permite a importação de um ou mais arquivos XML que contenham medições de métricas de software. É necessário que a ferramenta importe mais de um arquivo XML para um único sistema pois ele pode ser constituído de vários projetos. O formato do arquivo XML de entrada é o mesmo usado para a saída do *Eclipse Metrics Plugin*. Isso ocorre porque o *Eclipse Metrics Plugin* foi a ferramenta utilizada para extrair o *dataset* de métricas usado no processo de derivação dos valores referência, e também por ela ser uma ferramenta integrada com a IDE mais usada no mundo para desenvolvimento em Java, o Eclipse. Dessa forma, são evitadas as diferenças de medições de métricas de software que ocorrem por implementações distintas entre as ferramentas de extração e que geram ruído quando da utilização dos valores referência e a sua interpretação dos valores extraídos. O ideal é a utilização conjunta do *Eclipse Metrics Plugin* e da *RAFTool*, apesar de não ser um fato obrigatório. *RAFTool* permite, após a seleção do nome e a importação das medições, a opção de concluir aquela instanciação do sistema analisado. Nesse momento, a ferramenta processa os arquivos XML, inserindo as medições em um banco de dados *standalone*, que persistirá os dados para trabalhos de filtragem até o término daquela execução do programa. Esses dados não ficam persistidos para uma próxima execução.

4.2. Processo de Filtragem de Métodos, Classes ou Pacotes

O usuário deve selecionar o tipo de artefato filtrado: método, classe ou pacote. Posteriormente, ele deve digitar a expressão booleana de filtragem de métodos, classes ou pacotes. Essa expressão booleana deve ser uma combinação de ANDs e ORs dentro das faixas dos valores referência das métricas que serão utilizadas na filtragem que está sendo criada. Para isso, o seguinte formato é

definido: COMMON|CASUAL|UNCOMMON[METRIC_ID], onde COMMON corresponde à faixa *Bom/Regular*, CASUAL à faixa *Regular/Ocasional* e UNCOMMON à faixa *Ruim/Raro*. Quando o usuário escolhe a faixa CASUAL, o sistema retorna os métodos, classes e pacotes que estão na faixa *Regular/Ocasional* ou *Ruim/Raro*. METRIC_ID é o identificador da métrica que deve estar naquela faixa, devendo seguir o padrão estabelecido pelo Eclipse Metrics Plugin como ID. Logo, se o usuário digitar CASUAL[MLOC], por exemplo, o processo de filtragem retorna todos os métodos do sistema instanciado cuja classificação pelo valor referência sugerido para a métrica *Número de Linhas de Código por Método* (MLOC) for *Regular/Ocasional*. Com a utilização de ANDs e ORs, o usuário pode compor sua filtragem com vários valores referência. Por exemplo, se o usuário digitar CASUAL[MLOC] AND UNCOMMON[NBD], o processo de filtragem retorna todos os métodos do sistema classificados como *Regular/Ocasional* para a métrica *Número de Linhas de Código por Método* (MLOC) e, desses métodos, todos aqueles classificados como *Ruim/Raro* pela métrica *Profundidade de Blocos Aninhados* (NBD). Parênteses são aceitos nessa expressão como indicador da precedência das operações. *RAFTool* acessa o banco de dados criado na fase de instanciação e seleciona os artefatos conforme os parâmetros estabelecidos.

4.3. Visualização/Exportação dos Resultados

Os artefatos filtrados são exibidos em uma tabela. Para o caso dos métodos, são exibidos o nome do método, o nome da classe e o nome do pacote. Para as classes, são o seu nome, o nome do arquivo da classe e o nome do pacote. A exibição do nome do arquivo é útil pois um arquivo pode conter mais de uma classe. Para os pacotes, são exibidos somente os nomes. Os artefatos filtrados podem ser exportados para uma planilha em formato CSV, arquivo escolhido pelo usuário na hora da exportação.

5. Arquitetura

5.1. Model-View-Controller (MVC)

Para o desenvolvimento da ferramenta *RAFTool* foi utilizado o padrão arquitetural Model-View-Controller (MVC) [Sommervile 2010], que separa a apresentação e a interação da lógica de negócio e dados do sistema. O Modelo encapsula, além dos objetos de domínio, o acesso ao banco de dados utilizado pela aplicação. Outra responsabilidade do modelo dentro da *RAFTool* é relativo à lógica de negócio da aplicação, que são objetos cujas responsabilidades consistem em realizar a lógica relativa ao processo de filtragem. Esse processo de filtragem envolve a interpretação das expressões booleanas utilizadas pelos usuários, bem como a sua aplicação nos artefatos selecionados. Utilizou-se o *framework* Javaluator[3], que auxilia na avaliação de expressões regulares em Java. Na parte de persistência, foi utilizado o banco de dados HSQLDB. HSQLDB é um servidor de banco de dados embarcado, que foi desenvolvido na plataforma Java e é de código aberto.

A Visão foi desenvolvida com a utilização do *Swing*. *Swing* implementa um conjunto de componentes para construir interfaces gráficas para interação com usuários (GUIs). Os componentes *Swing* são

3 http://javaluator.sourceforge.net/en/home/

implementados diretamente em Java. Utilizou-se a IDE Netbeans, que possui recursos de construção de interfaces *Swing* com a utilização de *Drag and Drop*, facilitando a construção de interfaces gráficas.

O Controle tem a responsabilidade de mapear as ações dos usuários em ações da camada de Modelo. Além disso, mediante a resposta da Camada de Modelo, ele é responsável por selecionar a visão apropriada ao usuário. Se a interface for trocada, por exemplo, o Controle cuida para que a camada de Modelo se mantenha inalterada.

5.2. Estrutura de Pacotes da *RAFTool*

Nesta seção são documentados os pacotes do sistema, suas responsabilidades e como interagem entre si. A Figura 2 mostra o diagrama de pacotes de *RAFTool*, que exibe os módulos do sistema e as dependências entre eles, ilustrando a arquitetura proposta para o sistema. A seguir são descritas as responsabilidades de cada um desses pacotes:

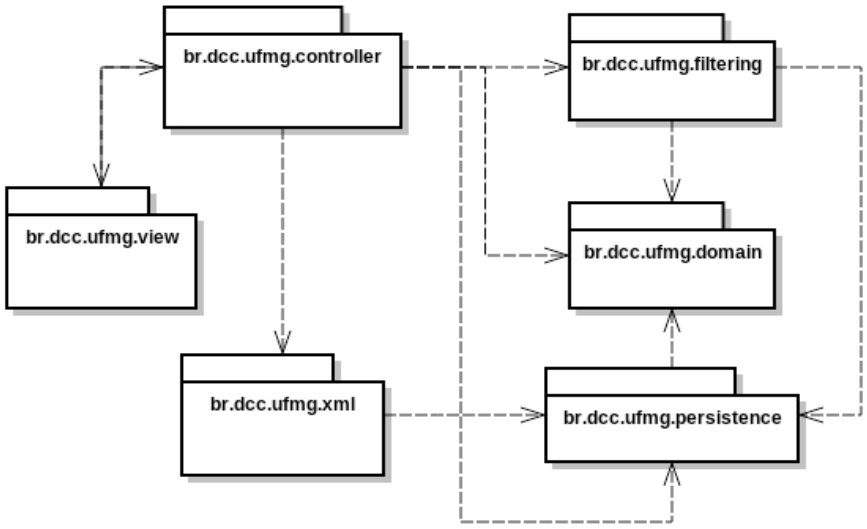

Figure 2. Diagrama de Pacotes da *RAFTool*.

1. *br.dcc.ufmg.view*: agrupamento das classes responsáveis pela camada de visão do sistema. Possui dependência com o módulo *controller*.

2. *br.dcc.ufmg.controller*: agrupamento das classes responsáveis pela camada de controle do sistema. Possui dependência com o a *view*, bem como com os restante dos módulos do sistema, que constituem a camada de modelo.

3. *br.dcc.ufmg.xml*: agrupamento das classes que trabalham com a leitura dos documentos XML, conversão para objetos Java e importação dos dados lidos desses documentos para o banco de dados. Para tal, possui dependência em relação ao módulo *persistence*, responsável pelas operações de banco de dados.

4. *br.dcc.ufmg.filtering*: agrupamento de classes responsáveis pelas operações de filtragem a serem realizadas nos artefatos do software analisado. Esse módulo do sistema possui dependência

com o módulo *persistence*, que é responsável por acessar os dados do sistema analisado. Possui dependência com o módulo *domain*, que modela o cenário real de um software orientado por objetos.

5. *br.dcc.ufmg.domain*: agrupamento de classes responsáveis por representar o vocabulário e descrever os conceitos chaves de um software orientado por objetos, bem como as relações entre as entidades estabelecidas. Sobre os objetos do domínio são executadas as operações de filtragem de dados.
6. *br.dcc.ufmg.persistence*: agrupamento de classes responsáveis por tratar operações de inserção, leitura e atualização dos dados utilizados no sistema. É um módulo independente, que modela o domínio do problema.

As principais classes do sistema foram documentadas com a utilização do *Javadoc* e disponibilizados em formato HTML por meio do seguinte *link:*

```
http://www.dcc.ufmg.br/~tfilo/raftool/doc
```

5.3. Infraestrutura para Instalação e Utilização da *RAFTool*

Para instalar e utilizar a *RAFTool*, é necessário um computador com a máquina virtual Java instalada, com versão igual ou superior a 1.6. O executável está disponível em:

```
http://www.dcc.ufmg.br/~tfilo/raftool/raftool.jar
```

6. Conclusão

A ferramenta RAFTool realiza a filtragem de métodos, classes e pacotes que possuem medições anômalas de métricas de softwares orientados por objetos segundo os valores referência sugeridos por Filó *et al.* (2014). Como exemplo da utilização da *RAFTool*, disponibilizamos um vídeo-tutorial *online* que mostra o uso da ferramenta:

```
http://www.dcc.ufmg.br/~tfilo/raftool/demo_raftool.htm
```

RAFTool é uma ferramenta que suporta a utilização desses padrões de forma efetiva na filtragem de métodos, classes e pacotes com medições anômalas. A ferramenta apoia de forma efetiva os Processos de Medição - MED do nível F (Gerenciado) do MR-MPS-SW, proporcionando a efetiva transformação de uma série de medidas associadas às estruturas internas do código-fonte em informações úteis sobre sua qualidade, gerando informações úteis para a tomada de decisão. Isso permite à organização direcionar esforços de forma pontual dentro do código-fonte para realizar, por exemplo, refatorações e maior carga de testes, haja visto o maior risco. Dessa forma, espera-se que, em conjunto com o *Eclipse Metrics Plugin* e os valores referência sugeridos por Filó *et al.*, *RAFTool* colabore com os esforços de pesquisa que visam a aplicar de forma efetiva as métricas

de software no gerenciamento da qualidade interna de software por meios quantitativos. Como trabalhos futuros, vislumbramos a integração da *RAFTool* ao *plugin Eclipse Metrics*, monitorando o código de forma dinâmica durante o desenvolvimento, manutenção e evolução de software, disparando alarmes que possibilitem aos desenvolvedores a utilização de métricas de forma integrada à ferramenta de desenvolvimento. Nessa ferramenta, o desenvolvedor não precisaria se preocupar com números, pois os valores referência propostos funcionariam como sinais sinais de trânsito: a faixa *Bom/Frequente* seria o sinal "verde", a faixa *Regular/Ocasional* o sinal "amarelo", sugerindo atenção, e a faixa *Ruim/Raro* o sinal "vermelho", indicando que deve-se parar e analisar o que está sendo codificado. O código-fonte da *RAFTool* está disponível no *GoogleCode*, que hospeda projetos em um ambiente colaborativo de desenvolvimento, sobre os termos da GPLv3. Utilizou-se o *Subversion* como ferramenta de controle de versão. O código-fonte pode ser baixado diretamente no Eclipse e está disponível por meio do seguinte *link*:

```
https://code.google.com/p/raftool/source/checkout
```

References

Filó, T., Bigonha, M., and Ferreira, K. (2014). Um método de extração de valores re- ferência para métricas de softwares orientados por objetos. In CBSoft-WTDSoft 2014.

Softex, X. (2012). MPS.BR - Melhoria de Processo do Software Brasileiro. Technical report, Association for Promoting the Brazilian Software Excellence - SOFTEX.

Sommerville, I. (2010). Software Engineering. Addison-Wesley, Harlow, England, 9 edition.

Softex

mps Br Melhoria de Processo do Software Brasileiro

O WAMPS 2014 (X Workshop Anual do MPS) tem por objetivo reunir a Comunidade de Prática do MPS, com foco nos implementadores, avaliadores, consultores de aquisição e instrutores MPS, membros da ETM/MPS - Equipe Técnica do Modelo e do FCC/MPS - Fórum de Credenciamento e Controle, IOGE/MPS – Instituições Organizadoras de Grupos de Empresas e Agentes Regionais Softex, clientes MPS e pessoal da Indústria, Governo, Academia do país e exterior, interessados tanto no aprimoramento de processos de software e serviços de TI usando o modelo MPS quanto nos resultados alcançados no programa MPS.BR – Melhoria de Processo do Software Brasileiro.

Este ano se comemora a décima edição do WAMPS e a superação da marca de 600 avaliações MPS publicadas, a programação desta nona edição do WAMPS conta com um palestrante internacional e palestrantes nacionais especialmente convidados, além de oferecer três cursos e dois painéis de grande interesse nesta área, acolher importantes reuniões como a reunião semestral do CGP/MPS.BR – Conselho de Gestão do Programa e realizar a entrega de placas para empresas recém-avaliadas MPS.

Além disso, com o apoio da SBC - Sociedade Brasileira de Computação, o WAMPS promoveu uma chamada de trabalhos envolvendo engenharia de software, que estejam relacionados ou sejam aplicáveis ao contexto de iniciativas de melhoria de processos de software e serviços. Os vinte e dois trabalhos aceitos serão apresentados em sete sessões técnicas. Nesta publicação encontram-se todos os artigos selecionados pelo Comitê de Programa do WAMPS 2014 para apresentação nas sessões técnicas.

apoio:
Finep INOVAÇÃO E PESQUISA
Ministério da Ciência, Tecnologia e Inovação
GOVERNO FEDERAL BRASIL PAÍS RICO É PAÍS SEM POBREZA

ISBN 978-85-99334-80-5
9 788599 334805

WWW.SOFTEX.BR/MPSBR