PRISCILA PEREIRA DE SOUZA

A UTILIDADE DOS VALORES REFERÊNCIA DE MÉTRICAS NA AVALIAÇÃO DA QUALIDADE DE SOFTWARES ORIENTADOS POR OBJETO

Proposta de dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADORA: Mariza Andrade da Silva Bigonha

COORIENTADORA: Kecia Aline Marques Ferreira

Belo Horizonte

Junho de 2015

Sumário

Capítulo 1

Introdução

Há um crescente interesse por parte das organizações em produzir software de maneira eficiente, com qualidade, de fácil entendimento e manutenção, a um baixo custo e em um período de tempo mínimo. Para alcançarem esses objetivos, uma das medidas adotadas é a utilização de práticas baseadas em objetos, simplificando o projeto de softwares complexos. Para Amber (1998), as organizações escolhem a orientação por objetos (OO) porque querem dar às suas aplicações mais qualidade, implementar sistemas seguros, com um menor custo e menor tempo.

Problemas na qualidade do software geralmente são identificados por meio de testes e de inspeções, técnicas nas quais os esforços de manutenção se concentram principalmente na correção de falhas [Eaddy et al.,2008] e na adição de novas funcionalidades [Conejero et al., 2012]. Porém, essas atividades demandam tempo e geram custos. A utilização de valores referência associado a métricas, cria uma alternativa para identificar falhas de softwares orientados por objeto e detectar desvios de projeto, conhecidos na literatura como *bad smells* [Fowler, 1999].

Segundo Pressman (2006), as métricas de softwares proporcionam medidas quantitativas para avaliar a qualidade dos projetos, permitindo aos engenheiros de software efetuarem alterações no decorrer do processo de desenvolvimento que irão melhorar a qualidade do produto final. Os principais objetivos das métricas de softwares são: prover um melhor entendimento da

qualidade do produto, avaliar a efetividade do processo e melhorar a qualidade do trabalho executado na fase de projeto. De acordo com Côrtes e Chiossi (2001), quando são calculadas métricas, pretende-se obter dados que irão proporcionar opções para uma melhoria.

Apesar da importância das métricas no gerenciamento da qualidade de softwares orientados por objetos, elas ainda não foram efetivamente utilizadas na indústria de software [Riaz et al., 2009; Tempero et al., 2010; Ferreira et al., 2012]. Uma possível razão é que, para a grande maioria das métricas, não são conhecidos os valores referência [Ferreira et al., 2012]. De acordo com diversos autores, existe um campo de pesquisa em aberto na definição de métodos para derivação de valores referência para métricas de software orientado por objetos [Rosenberg et al.,1999; Benlarbi et al., 2000; Shatnawi et al., 2010; Chikara et al., 2011; Ferreira et al., 2011, 2012; Kaur et al., 2013; Oliveira et al., 2014, Filó et al., 2014]. Como consequência, esse tem sido o foco principal dos trabalhos realizados nessa área até então. De acordo com Filó et. al (2014) há uma quantidade razoável de métodos definidos para derivação de valores referência e dezenas de métricas propostas na literatura, porém, há poucas métricas com valores referência conhecidos, dificultando e até mesmo impossibilitando o gerenciamento da qualidade de software por meios quantitativos.

Para contornar esse problema, alguns autores têm investido em definir valores referência para métricas de software como por exemplo: Filó et al., 2014, Ferreira et al., 2011, 2012 e Couto, et al.,2013. Embora os autores desses trabalhos tenham avaliado suas propostas, tais avaliações não são abrangentes. Filó (2014), por exemplo, avaliou apenas alguns *bad smells* e Ferreira (2012) avaliou a aplicabilidade de valores referência apenas na identificação de desvios de projeto. É necessário um estudo empírico mais abrangente e detalhado para investigar a aplicabilidade de valores referências em cenários reais de Engenharia de Software, por exemplo, na identificação de *bad smells* e na predição de falhas.

Dado esse cenário, a proposta dessa dissertação consiste em: investigar a utilidade dos valores referência na avaliação da qualidade de softwares orientados por objeto; avançar no estado da arte analisando uma maior quanti-

dade de métricas com valores referência; compreender e estabelecer possíveis relações entre valores referência de métricas, falhas e *bad smells* no que diz respeito a qualidade de software.

Para mostrar nossos resultados, nós propomos responder, separadamente, as seguintes questões de pesquisa:

1. Os valores referência de métricas de software orientados por objetos podem identificar falhas?

2. Os valores referência de métricas de software orientados por objetos podem detectar *bad smells*?

1.1 Objetivos

O objetivo geral deste trabalho consiste em realizar um estudo empírico a fim de verificar a utilidade dos valores referência de métricas de software propostos na literatura para avaliação da qualidade de softwares orientados por objeto.

Como objetivos específicos pretende-se investigar se via valores referências das métricas é possível realizar a predição de falhas e a detecção de *bad smells*.

1.2 Contribuição Pretendida

A principal contribuição esperada para esse trabalho de mestrado consiste na avaliação dos valores referência em softwares livres com o objetivo de verificar se, de fato, é possível por meio deles realizar a predição de falhas e a detecção de *bad smells* em softwares orientados por objeto. Espera-se que no fim desse trabalho de pesquisa, os resultados obtidos possam ser úteis para um uso mais efetivo dos valores referência no gerenciamento da qualidade de software por meios quantitativos.

1.3 Estrutura da Proposta de Dissertação

O restante desta proposta de dissertação está organizada da seguinte forma:

Capítulo 2 apresenta uma revisão preliminar de literatura sobre a importância da medição de software, métricas de softwares orientados por objetos, valores referência de métricas, *bad smells* e predição de falhas.

Capítulo 3 apresenta os pontos principais da metodologia que será empregada para o desenvolvimento desse trabalho de dissertação.

Capítulo 4 apresenta o planejamento para a realização do trabalho. Inicialmente mostra o cronograma proposto para elaboação da dissertação, seguido de uma proposta de sumário da dissertação.

Capítulo 2

Revisão de Literatura

Esse capítulo apresenta uma revisão bibliográfica sobre medição de software, métricas de softwares orientados por objetos, valores referência para métricas de softwares orientados por objetos, predição de falhas e *bad smells*.

2.1 Medição de Software

A preocupação com a qualidade dos produtos tem se tornado requisito imprescindível devido ao mercado estar cada vez mais competitivo, e ao fato de que organizações que disponibilizam produtos com baixa qualidade podem rapidamente perder seus clientes. Por essa razão, as empresas tem adotado novas técnicas e tecnologias, como o uso de desenvolvimento orientado por objetos e de ferramentas CASE. Além disso, atualmente há uma conscientização da importância do gerenciamento de qualidade de software e da utilização de técnicas de gerenciamento de qualidade provenientes da necessidade de realizar manutenção no software [Sommerville, 2010]

Um elemento fundamental para qualquer processo que objetive atingir qualidade em seu produto final é a medição. Medidas são utilizadas para que os modelos criados tenham sua qualidade avaliada e, consequentemente, para que o padrão de qualidade estabelecido como aceitável seja atingido no fim do processo. Atualmente, a medição é vista como um trabalho pragmático com o objetivo de encontrar o indicador direto para um determinado aspecto da

qualidade. No entanto, sem critérios claros, as medições podem impedir comparações úteis com outros projetos. Portanto, métricas têm de ser escolhidas por meio de uma referência clara e confiável. Elas são experimentalmente avaliadas para medir um atributo de qualidade [Baggen et al., 2010].

2.1.1 Métricas de Software

Medição é parte fundamental de qualquer disciplina de engenharia, e disciplinas da Engenharia de Software não são exceções. Sommerville (2010) define medição como um valor numérico relacionado a algum atributo de software, permitindo a comparação entre atributos da mesma natureza. Ele argumenta que a medição permite realizar previsões gerais de um sistema e identificar componentes com anomalias.

As métricas de software referem-se a medições que podem ser aplicadas para verificar os indicadores de processos, projetos e produtos de software. As métricas possibilitam medições de padrões para avaliar determinados atributos que estejam relacionados ao software. De acordo com Sommerville (2010) por meio das métricas é possível gerenciar um software. É possível também, por exemplo, via atributos internos de um software, definir atributos externos como facilidade de manutenção, confiabilidade, portabilidade e facilidade de uso.

As métricas também podem ser usadas para identificar problemas estruturais nas fases iniciais do ciclo de vida do software e na identificação dos desvios de projeto, conhecidos na literatura como *bad smells* [Fowler, 1999]. Alguns trabalhos têm sido desenvolvidos para identificar *bad smells* em software com o uso de métricas a partir de código fonte [Marinescu, 2002; Lanza et al.,2006], e no início do projeto de desenvolvimento por meio de diagramas de classes [Nunes,2014].

Existem diversas métricas para orientação por objetos propostas na literatura, dentre as quais destacam-se:

- Métricas CK: propostas por Chidamber e Kemerer (1994), elas avaliam o software orientado por objetos no nível de classe. São compostas por seis métricas: Métodos Ponderados por Classe (WMC), Profundidade

de Árvore de Herança (DIT), Número de Filhos (NOC), Acoplamento entre Classes de Objetos (CBO), Resposta de Classe (RFC), Ausência de Coesão em Métodos (LCOM).

Os autores desse conjunto de métricas afirmam que essas métricas podem auxiliar os usuários no entendimento da complexidade de projetos orientados por objetos, na detecção de anomalias e na previsão de alguns resultados do projeto. Tais métricas foram propostas com o intuito de avaliar atributos de qualidade externa do software, tais como: defeitos de software, testes e esforço de manutenção [Basili et al., 1996; Chidamber e Kemerer, 1994].

- Métrica MOOD: propostas por Brito e Abreu e Carapuca (1994), são compostas por oito métricas de software orientado por objetos, que avaliam aspectos de herança, encapsulamento, coesão, acoplamento, polimorfismo e reúso de software. O conjunto MOOD é composto das seguintes métricas: Fator Ocultação de Método, Fator Ocultação de Atributo, Fator Herança de Método, Fator Herança de Atributo, Fator Acoplamento, Fator Polimorfismo, Fator Reúso.

  O conjunto MOOD tem o objetivo de proporcionar indicadores quantitativos para as propriedades dos projetos orientados por objeto via métricas [Pressman, 2006].

- Métricas MARTIN: Martin (1994) propõe um conjunto de métricas para medir a qualidade de softwares orientados por objetos em termos de acoplamento. Projetos com alto grau de acoplamento tendem a ser rígidos, não reutilizáveis e de baixa qualidade em relação a manutenabilidade. O conjunto é composto pelas seguintes métricas: Acoplamento Aferente (AC), Acoplamento Eferente (EC), Instabilidade, Abstração e Distância Normalizada.

  De acodo com Sommerville (2010), avaliar a qualidade do software por meio de medições possibilita definir quantitativamente o sucesso ou a falha de determinado atributo, identificando a necessidade de melhorias. Gerenciar a qualidade do software pode permitir que sejam alcançados

um baixo número de defeitos e padrões controláveis de manutenabilidade, confiabilidade e portabilidade.

Apesar da importância das métricas no gerenciamento da qualidade de softwares orientados por objetos, elas ainda não foram efetivamente utilizadas na indústria de software [Riaz et al., 2009; Tempero et al., 2010; Ferreira et al., 2012, Filó et. al.,2014 e Filó et. al.,2015]. Uma possível razão é que para a grande maioria das métricas, não são conhecidos valores referência [Ferreira et al., 2012]. Alguns trabalhos têm sido realizados recentemente para contornar esse problema. Esse tópico é tratado na Seção 2.2.

2.2 Valores Referência

Valores referência são valores obtidos por observação ou mensuração que auxiliam na avaliação da qualidade do software.

Com a utilização de valores referência para métricas de software orientado por objetos, possíveis problemas no software poderiam ser identificados durante o projeto, resultando em um software de maior qualidade e, consequentemente, com um menor custo de manutenção. Além disso, informações acerca da qualidade interna do software poderiam ser obtidas e direcionar esforços de testes e manutenções preventivas para pontos críticos do sistema, otimizando a aplicação dos recursos disponíveis durante a fase de operação [Baxter et al., 2006; Riaz et al., 2009; Tempero et al., 2010; Ferreira et al., 2012].

Na literatura, encontramos alguns trabalhos que vem sendo realizados com o objetivo de derivar valores referência para métricas [Rosenberg et al., 1999; Benlarbi et al., 2000; Shatnawi et al., 2010; Chhikara et al., 2011; Ferreira et al., 2012, 2011; Kaur et al., 2013; Oliveira et al., 2014; Filó et. al, 2014]. Esses trabalhos variam principalmente na metodologia de pesquisa utilizada para estabelecer esses valores, que de acordo com Filó et. al (2014) podem ser categorizadas da seguinte forma:

- Pesquisa Quantitativa: utilizam técnicas estatísticas por meio das se-

guintes estratégias:

  - Regressão Logística: utiliza modelos de regressão logística com o intuito de identificar grupos de risco, que são classes fora dos valores definidos, a partir de um conjunto de dados constituídos por métricas.
  - Análise de Distribuição: baseiam sua análise em distribuições estatísticas da coleta de medições de um grande número de softwares.
  - Análise ROC (*Receiver-Operating Characteristic*): utiliza uma representação gráfica para ilustrar e avaliar a correta avaliação da classe em relação à presença de erros com a variação do valor referência.

- Pesquisa Qualitativa: avaliam projetos propostos para um mesmo problema, partindo de modelagens sabidamente ruins a boas para identificar a evolução das medições das métricas.

Com a aplicação das métricas em conjunto com um catálogo de valores referência, as avaliações podem ser realizadas de forma automatizada, baseada em aspectos quantitativos do software, auxiliando na garantia da qualidade.

Avaliar a eficácia da utilização dos valores referência propostos na literatura é um ponto importante de ser investigado. Em sua maior parte, os valores referência propostos não foram avaliados amplamente, que é de fundamental importância para a aplicação desses valores referência na prática.

2.3 *Bad Smell*

De acordo com Martin Fowler, o termo *bad smell* (mau cheiro) se refere às características encontradas nas estruturas de códigos que indicam que eles estão com problemas e precisam ser refatorados de alguma maneira. Os *bad smells* não são a causa direta de falhas na aplicação, mas podem influenciar indiretamente para a inserção de erros responsáveis por futuras falhas [Fowler, 1999]. Em geral, eles são responsáveis pelas dificuldades de manutenção

e evolução do sistema, realização de testes e propicia a inserção de *bugs* [Mansoor et al., 2014]. Por essa razão é essencial saber como identificá-los para se aplicar os mecanismos necessários para sua remoção e, consequentemente, melhorar a qualidade do software.

Martin Fowler em seu livro (1999), expõe que a identificação de *bad smell* é o primeiro passo para realização de refatorações controladas e em pequenos passos. São exemplos de *bad smells*: *Long Method* (Método Longo), *Large Class* (Classes Grande), *Long Parameter List* (Lista grande de parâmetros), *Divergent Change* (Mudança Divergente), *Duplicated Cod* (Código Duplicado), *Comments* (comentários), dentre outros.

É importante que a remoção dos *bad smells* seja realizada o mais cedo possível no desenvolvimento do software. Entretanto, a identificação deles pode ser fragilizada, pois dependem diretamente da interpretação e habilidade de identificação do desenvolvedor. Outro problema é não conhecer quais os mecanismos podem ser aplicados para a remoção desses *bad smells* de código. As métricas podem auxiliar a identificar os *bad smells*, mas para isso, é necessário identificar algo que os relacione. Esse relacionamento é possível via valores referência, que definem valores numéricos que determinam quantitativamente se o valor de uma métrica é aceitável.

O trabalho de Marinescu (2002) é possivelmente um dos primeiros a definir estratégias de detecção de *bad smells* utilizando métricas. Lanza et al. (2006) também definem uma lista de *bad smells*, alguns novos e outros modificados a partir da obra de Fowler (1999), e usam estratégias de detecção para identificação deles. Em seu trabalho sobre refatoração, Hamza et al. (2008) descrevem os *bad smells* das obras de Fowler (1999) e Kerievsky (2005). Outros trabalhos propõem a detecção de *bad smell* por meio de diagramas UML e diagramas de classes [Nunes,2014; Bertran, 2009] e a detecção de *bad smells* por meio de métricas de interesse [Padilha et. al.,2012].

Outro trabalho realizado nessa área é o de DAmbros et al. (2010) que consideram que a maior parte dos problemas de software estão relacionados a manutenção e que métricas somente serão efetivas se forem combinadas formando estratégias de detecção. Esses autores propõem relacionar *bad smells* com falhas de software.

2.4 Predição de Falhas de Software

Sommerville (2010) define falha de software como a incapacidade do software de realizar uma função requisitada (aspecto externo). Predição de falhas é uma área de pesquisa em Engenharia de Software que objetiva identificar os componentes de um sistema de software que são mais prováveis de apresentar defeitos [Cesar, et. al, 2013].

As técnicas de predição de defeitos envolvem a análise do código-fonte de uma versão de um sistema [Basili et al. 1996, Marcus et al. 2008] ou de dados históricos do controle de versão [Moser et al. 2008, Nagappan et al. 2010] e de sistemas de acompanhamento de defeitos [Kim et al. 2007]. A finalidade é prever a ocorrência ou o número de defeitos que serão encontrados em cada componente do sistema, por exemplo: pacote, classe, método ou qualquer outra unidade de código [DAmbros et al. 2010].

No trabalho de Couto (2013) é proposta uma abordagem para predição de falhas centrada em evidências de causalidade entre métricas de código fonte (como preditor) e a ocorrência de defeitos. Mais especificamente, foi utilizado um teste de hipótese estatístico para avaliar se variações passadas nos valores de métricas de código fonte podem ser usados para predizer mudanças em séries temporais de defeitos. Porém, Couto (2013) não utilizou valores referências de métricas.

Pretende-se com este trabalho de dissertação pesquisar mais referências na literatura sobre técnicas para predição de falhas em softwares e investigar se por meio de métricas é possivel realizar a predição de falhas com o apoio de valores referências.

2.5 Análise Crítica

Há dezenas de métricas propostas na literatura onde não são conhecidos valores referência para elas, e isso é um impedimento para a sua aplicação nas fábricas de software. Nos trabalhos relacionados encontrados, podemos observar que os valores referência propostos e avaliados por seus autores apresentam limitações, pois eles não foram comparados entre si, ou não foram

aplicados para um número suficiente de métricas. Esses problemas precisam ser contornados por meio de uma avaliação sistemática e ampliando o leque de valores referência. Este trabalho de dissertação se propõe a contribuir com a solução para esse problema.

Capítulo 3

Trabalho Proposto

Este capítulo apresenta a metodologia, o cronograma e o sumário do trabalho de mestrado proposto.

3.1 Metodologia

O desenvolvimento desse trabalho de mestrado será composto de cinco etapas relacionadas a seguir:

Na primeira etapa será realizada uma revisão bibliográfica mais abrangente sobre qualidade de software, métricas de softwares orientados por objetos, valores referência para métricas de softwares orientados por objetos, métodos para a predição de falhas e identificação de *bad smells*. Conforme mencionado por Jung (2004), a pesquisa bibliográfica tem por finalidade principal formar uma consistente base mental a partir daquilo que é existente, e oportunizar uma ampla aquisição de conhecimentos para o entendimento substancial do assunto, viabilizando ao pesquisador ousar ao propor novos argumentos que justifiquem as descobertas.

Após a conclusão da revisão bibliográfica, na segunda etapa será realizado um planejamento dos experimentos a fim de identificar os softwares, as métricas, os valores referência e os *bad smells* que serão utilizados. Será realizada também, uma descrição dos processos que envolvem a preparação e condução do estudo experimental, a definição do contexto e formulação das

hipóteses relacionadas ao experimento.

Na terceira etapa, será investigada a necessidade de construção de uma ferramenta de coleta de dados para obter as métricas dos softwares selecionados. Um experimento será realizado na quarta etapa e os resultados obtidos serão avaliados para verificar a possibilidade de identificar falhas e *bad smells* a partir dos valores referências de métricas.

A quinta etapa diz respeito a escrita de artigos e a finalização do texto da dissertação, que estará sendo produzido ao longo do desenvolvimento do trabalho proposto.

3.2 Cronograma

A Figura 3.1 apresenta o cronograma de tarefas proposto para as atividades de pesquisa a serem realizadas para a dissertação de mestrado.

Atividades	2015								2016						
	Maio	Junho	Julho	Agosto	Setembro	Outubro	Novembro	Dezembro	Janeiro	Fevereiro	Março	Abril	Maio	Junho	Julho
Definição do Problema															
Definição da Proposta															
Revisão Bibliográfica															
Planejamento dos experimentos															
Construção da Ferramenta de Coleta de Dados															
Coleta e análise dos resultados obtidos															
Escrita da dissertação															
Escrita de artigo															
Defesa da dissertação															

Figura 3.1: Cronograma

3.3 Proposta de Sumário para a Dissertação

O objetivo desta seção é ilustrar a estrutura que se pretende adotar na dissertação, por meio da construção do sumário.

1. Introdução
    - 1.1 Definição do Problema
    - 1.2 Objetivos
    - 1.3 Contribuições
    - 1.4 Organização do Trabalho
2. Medição de Software
    - 2.1 Métricas de Software Orientado por Objetos
        - 2.1.1 Métricas CK
        - 2.1.2 Métricas MOOD
        - 2.1.3 Métricas de Martin
        - 2.1.4 Análise Crítica
3. Valores Referência para Métricas de Software Orientados por Objetos
    - 3.1 Pesquisa Quantitativa
    - 3.2 Pesquisa Qualitativa
    - 3.3 Resumo dos Trabalhos Estudados
    - 3.4 Análise Crítica
4. Falhas de Software
    - 4.1 Métricas para predição de falhas
    - 4.2 Análise Crítica
5. Bad Smell
    - 5.1 Metricas para detecção de bad smell

Bibliografia Referênciada e a Estudar

Abreu, F. B. and Carapuça, R. (1994). Object-oriented software engineering: Measuring and controlling the development process. Em proceedings of the 4th International Conference on Software Quality, volume 186.

Basili, V. R.; Briand, L. C. and Melo, W. L. (1996). A validation of object-oriented design metrics as quality indicators. IEEE Trans. Softw. Eng., 22:751?761.

Bertran, I. M. (2009). Avaliação da qualidade de software com base em modelos uml. Dissertação da PUC-RJ. Rio de Janeiro, RJ, Brasil. Pontifícia Universidade Católica do Rio de Janeiro.

Chhikara, A.; Chhillar, R. S. and Khatri, S. (2011). Evaluating the impact of diferent types of inheritance on the object oriented software metrics. Internation Journal of Enterprise Computing and Business Systems, 1(2). ISSN 22308849-V1I2M7-072011.

Chidamber, S. R. and Kemerer, C. F. (1994). A metrics suite for object oriented design. IEEE Trans. Softw. Eng., 206476493. ISSN 0098-5589.

Conejero, J. M.; Figueiredo, E.; Garcia, A.; Hernandez, J. and Jurado, E. (2012). On the relationship of concern metrics and requirements maintainability. Inf. Softw. Technol., 54(2):212?238.

Couto, C.;Bigonha, R.;Valente M; Anquetil, N. Predizendo defeitos de software com testes de causalidade. Tese de Doutorado, DCC/UFMG.

DAmbros, M.; Bacchelli, A. and Lanza, M. (2010). On the impact of design flaws on software defects. Em Quality Software (QSIC), 2010 10th International Conference on, pp. 23-31. IEEE.

Ferreira, K. A. M. (2011). Um modelo de predição de amplitude da propagação de modificações contratuais em software orientado por objetos. Tese de doutorado, DCC/ UFMG.

Ferreira, K. A.; Bigonha, M. A.; Bigonha, R. S.; Mendes, L. F. and Almeida, H. C. (2012). Identifying thresholds for object-oriented software metrics. Journal of Systems and Software, 85(2):244–257. ISSN 01641212

Figueiredo, E.; Padilha, J.(Detecção de anomalias de código usando métricas de software). Dissertação de Mestrado, DCC/UFMG.

Filó,T;Bigonha, M.;Ferreira, K.(2014). Identificação de valores referência para métricas de softwares orientados por objetos.

Fowler, M. (1999). Refactoring: Improving the Design of Existing Code. Addison- Wesley, Boston, MA, USA. ISBN 0-201-48567-2.

Kerievsky, J. (2005). Refactoring to patterns. Pearson Deutschland GmbH.

Lanza, M.; Ducasse, S. and Marinescu, R. (2007). Object-Oriented Metrics in Practice: Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object- Oriented Systems. Springer. ISBN 9783540395386.

Marinescu, R. (2002). Measurement and quality in object-oriented design.

Marinescu, R. (2004). Detection strategies: Metrics-based rules for detecting design flaws. Em In Proc. IEEE International Conference on Software Maintenance

Martin, R. (1994). OO design quality metrics - an analysis of dependencies. Em Workshop Pragmatic and Theoretical Directions in Object-Oriented Software Metrics . OOPSLA'94

Nunes, H; Bigonha, M; Ferreira, K.(2014). Identificação de bad smell em software a partir de modelos UML. Dissertação de Mestrado, DCC/UFMG.

Pressman, R. S. (2006). Engenharia de Software. MacGraw Hill, Rio de Janeiro, 6º edição.

Riaz, M.; Mendes, E. and Tempero, E. D. (2009). A systematic review of software maintainability prediction and metrics. Em ESEM, pp. 367377.

Rosenberg, L.; Ruth, S. and Gallo, A. (1999). Risk-based Object Oriented Testing. Em In: Proceedings of the 24 th annual Software Engineering Workshop, NASA, Software Engineering Laboratory.

Sommerville, I. (2010). Software Engineering. Addison-Wesley, Harlow, England, 9 edição. ISBN 978-0-13-703515-1.

Tempero, E.; Anslow, C.; Dietrich, J.; Han, T.; Li, J.; Lumpe, M.; Melton, H. and Noble, J. (2010). Qualitas corpus: A curated collection of java code for empirical studies. Em 2010 Asia Pacific Software Engineering Conference (APSEC2010), pp. 336-345.