IMPACT OF SOFTWARE METRICS ON MAINTENANCE INDEX: A SYSTEMATIC REVIEW

IMPACTO DAS MÉTRICAS DE SOFTWARE NO ÍNDICE DE MANUTENÇÃO: UMA REVISÃO SISTEMÁTICA

ÁVILA, Allana; CARVALHO, Marcos Alberto; CARVALHO, Jaqueline Corrêa Silva; SANTOS, Flávia Aparecida Oliveira; RAMOS, Celso de Ávila; SILVA, Vinícius Duarte Esteves; SOUZA, Patrícia Carolina; BASTOS, Camila

Allana Ávila, UNIFENAS, Brasil

Marcos Alberto Carvalho, UNIFENAS, Brasil

Jaqueline Corrêa Silva Carvalho, UNIFENAS, Brasil

Flávia Aparecida Oliveira Santos, UNIFENAS, Brasil

Camila Bastos, UNIFENAS, Brasil

Patrícia Carolina Souza, UNIFENAS, Brasil

Celso de Ávila Ramos, UNIFENAS, Brasil

Vinícius Duarte Esteves Silva, UNIFENAS, Brasil

Revista Científica da UNIFENAS
Universidade Professor Edson Antônio Velano, Brasil
ISSN: 2596-3481
Publicação: Trimestral
vol. 6, nº. 5, 2024
revista@unifenas.br

Recebido: 08/07/2024
Aceito: 28/08/2024
Publicado: 09/09/2024

URL: https://revistas.unifenas.br/index.php/revistaunifenas/issue/view/52

DOI: 10.29327/2385054.6.5-8

ABSTRACT: This article explores the importance of software quality, highlighting its ability to meet users' needs and add value to the product. The research aims to investigate and define methods for calculating the impact of software metrics on the Maintenance Index of object-oriented projects. A Systematic Literature Review (SLR) was conducted to identify the main quality metrics used and how they affect software maintainability. The analysis of the 29 selected articles revealed that metrics such as cyclomatic complexity, cohesion, and coupling have a significant influence on the Maintenance Index. The main conclusion points to the need to integrate these metrics into automated tools to facilitate continuous and proactive assessment of software quality throughout its lifecycle, contributing to more efficient development practices.

KEYWORDS: Software Quality, Software Metrics, Maintenance Index, Systematic Literature Review, Software Measurements.

RESUMO: Este artigo explora a importância da qualidade do software, destacando sua capacidade de atender às necessidades dos usuários e agregar valor ao produto. A pesquisa objetiva investigar e definir métodos para calcular o impacto das métricas de software no Índice de Manutenção de projetos orientados a objetos. Para isso, foi realizada uma Revisão Sistemática da Literatura (RSL), com o intuito de identificar as principais métricas de qualidade utilizadas e como elas impactam a manutenibilidade do software. A análise dos 29 artigos selecionados revelou que métricas como complexidade ciclomática, coesão e acoplamento têm influência significativa no Índice de Manutenção. A conclusão principal aponta para a necessidade de integrar essas métricas em ferramentas automatizadas para facilitar a avaliação contínua e proativa da qualidade do software ao longo do seu ciclo de vida, contribuindo para práticas de desenvolvimento mais eficientes.

PALAVRAS-CHAVE: Qualidade de Software, Métricas de Software, Índice de Manutenção, Revisão Sistemática de Literatura, Medições de Software.

1 INTRODUÇÃO

A qualidade do software é um fator crucial para o sucesso e aceitação de produtos e serviços no mercado atual. Garantir que um software atenda às necessidades e expectativas dos usuários, bem como dos demais stakeholders, é essencial para agregar valor ao produto e obter vantagem competitiva. Nesse contexto, a medição e a avaliação da qualidade interna do software, especialmente a partir do código-fonte, desempenham um papel fundamental para assegurar a qualidade geral do produto [1].

Métricas de software têm se mostrado ferramentas valiosas na Engenharia de Software, proporcionando medidas quantitativas para avaliar a qualidade dos projetos em diferentes fases de desenvolvimento. Essas métricas permitem não apenas entender o nível de qualidade do software, mas também avaliar a capacidade do processo de desenvolvimento e identificar áreas que necessitam de melhorias. Ao longo do ciclo de vida do software, essas medições permitem que as equipes realizem ajustes contínuos para garantir a entrega de um produto confiável e alinhado às expectativas dos usuários [2].

Entretanto, ao se tratar de projetos orientados a objetos, surgem desafios adicionais. A diversidade e a complexidade das métricas aplicáveis a esses projetos frequentemente resultam em dificuldades na obtenção de valores de referência que sejam amplamente aceitos. Essa situação pode limitar o uso eficaz das métricas, dificultando a avaliação da manutenibilidade e comprometendo a tomada de decisões ao longo do processo de manutenção [2][3].

Esse cenário reflete as dificuldades sobre como as métricas de software podem ser utilizadas adequadamente para garantir uma manutenção eficaz e contínua, especialmente em projetos onde a qualidade do código e a facilidade de manutenção são críticas para o sucesso a longo prazo. Para explorar as interações entre essas métricas e o Índice de Manutenção, este estudo procura executar uma Revisão Sistemática de Literatura para identificar as principais métricas de qualidade existentes e como elas impactam a manutenibilidade do software

Na próxima seção, será detalhada a metodologia utilizada para a pesquisa e para o alcance dos objetivos propostos. Em seguida, serão apresentados os resultados obtidos, as análises e as discussões relevantes, bem como as contribuições deste estudo para a área de Engenharia de Software.

2 METODOLOGIA

A Revisão Sistemática da Literatura é uma abordagem metodológica que permite identificar, analisar e sintetizar os documentos disponíveis na literatura científica sobre um tema específico. Uma RLS pode ser realizada para explorar as pesquisas já existentes sobre o uso de métricas de software na avaliação da qualidade de projetos orientados a objetos. As questões de pesquisa definidas para guiar a seleção de trabalhos relevantes foram:

“Quais métricas de qualidade impactam no Índice de Manutenção de Software Orientado à Objeto?”,

“Como elas impactam? “, “Como elas são calculadas?”.

Com base na questão, uma string de busca foi definida para auxiliar na padronização da pesquisa no repositório de trabalhos científicos. A string é composta por palavras-chave relevantes para o assunto associado ao artigo:

"Abstract”: “software quality", "software quality assurance", "software metrics", "software measurements", "maintenance index".

O nome e o endereço eletrônico do repositório utilizado estão listados na Tabela 1. A opção de utilizar um único repositório de pesquisa é fundamentada em sua especialização na área de Engenharia de Software e de métricas de qualidade, oferecendo uma ampla gama de fontes relevantes, como revistas científicas e conferências [4]. Essa escolha visa garantir foco e eficiência na busca por artigos de alta qualidade, otimizando os recursos disponíveis para a pesquisa.

Tabela 1. Repositórios de artigos científicos utilizados

Repositórios	Endereços Eletrônicos
IEEE Xplore Digital Library	http://ieeexplore.ieee.org

Para serem selecionados, os artigos deverão possuir acesso livre ao seu conteúdo e serem publicados a partir do ano 2000 (critérios de seleção). Não foram definidas restrições quanto ao idioma do artigo. Após a definição do protocolo, a seleção dos artigos relevantes foi realizada por meio da execução dos seguintes passos:

• Passo 1: Foi definida uma string de busca específica, utilizada no repositório selecionado para coletar artigos de conferências, de revistas publicadas no período de 2000 a 2023. Após a busca, os trabalhos foram analisados, retirando aqueles que não atendiam a esses critérios.

• Passo 2: Em seguida, foi realizada a seleção dos trabalhos relevantes por meio de leitura do título, do resumo e das palavras-chave dos trabalhos obtidos na busca.

• Passo 3: Os artigos resultantes na primeira seleção (Passo 2) foram reunidos em um único conjunto, facilitando o processo de análise para a verificação dos trabalhos duplicados.

• Passo 4: Foi realizada a seleção secundária. Nos artigos resultantes do Passo 3, foi realizada a releitura do resumo e a leitura das considerações finais elaboradas de cada artigo.

A quantidade de artigos obtidos com a execução desses passos é apresentada na Tabela 2. A segunda coluna contém a Quantidade Inicial (QI) de artigos obtidos após a realização do Passo 1. A terceira coluna contém a quantidade de artigos obtidos após a Seleção Primária (SP), em que foi realizada a leitura de títulos, dos resumos e das palavras-chave. Na quarta, na quinta e na sexta colunas, são apresentados os resultados obtidos com Seleção Secundária (SS), em que foi realizada a releitura do resumo e a leitura das conclusões. É apresentada a quantidade de artigos irrelevantes (IR), de

artigos repetidos (RP), de artigos incompletos (IN) e de artigos selecionados em cada repositório (R).

Tabela 2. Quantidade de artigos selecionados

Repositórios	QI	SP	SS			
			IR	RP	IN	R
IEEE Xplore	2699	54	25	0	0	29

Os 29 artigos selecionados foram utilizados para realizar a análise qualitativa e a análise quantitativa descritas nas próximas seções.

2.1 Análise qualitativa

O software orientado a objetos possui a proteção necessária para garantir um funcionamento eficiente, estabelecendo conexões entre várias subclasses. No entanto, é crucial ressaltar que a vedação entre classes é uma característica de um sistema de software bem planejado. Dessa forma, a orientação, a exibição e a análise dos resultados das proporções das classes de software tornou-se cada vez mais importante no campo.

Com leitura dos artigos selecionados na RSL, foram encontradas métricas de qualidade que impactam no Índice de Manutenção do Software Orientado a Objeto (OMI).

• Coesão: a coesão é um grau em que os métodos dentro de uma classe estão relacionados entre si. Com isso, quanto maior a coesão, menor a probabilidade de que um erro de um elemento afete outro elemento do módulo. [12]

• Acoplamento: o acoplamento avalia o estado entre os módulos de um sistema. Com isso, quanto maior o acoplamento, maior a probabilidade de que uma mudança do módulo afete outros módulos. [12]

• Complexidade: a complexidade avalia a dificuldade de entendimento de um módulo. Com isso, quanto maior a dificuldade, maior a probabilidade de que uma mudança no módulo obtenha erros. [12]

• Abstração: a abstração é um ato de capacidade de um módulo que apresenta conceitos essenciais, ignorando detalhes irrelevantes. Com isso, quanto maior a abstração, menor a probabilidade de que uma mudança no módulo afete outros módulos. [2][12]

• Reusabilidade: a reusabilidade avalia a capacidade de um módulo que pode ser utilizado em diferentes partes de um sistema. Com isso, quanto mais pode ser reutilizado no módulo, menor a probabilidade de que uma mudança nesse módulo afete outros módulos. [2]

• Testabilidade: a testabilidade avalia a facilidade de testar um módulo. Com isso, é a métrica que mede a facilidade com que um módulo ou método pode ser testado para detectar erros ou bugs. [2]

A análise do Índice de Manutenção de Software Orientado a Objeto (OMI) é influenciada diretamente pelas métricas acima, sendo que a importância de cada métrica pode variar conforme o contexto do projeto. A Tabela 3 oferece uma análise detalhada dessas métricas, sendo que cada métrica é acompanhado de sua fórmula, uma descrição e o impacto estimado no Índice de Manutenção.

3 RESULTADOS E DISCUSSÃO

Em relação às questões de pesquisa, foi possível extrair os seguintes resultados após a leitura e análise dos artigos da Revisão Sistemática de Literatura (RSL).

Quais métricas de qualidade impactam no Índice de Manutenção de Software Orientado a Objeto?

Tabela 3. Métricas de Qualidade

Métrica	Formula	Descrição	Impacto no índice de manutenção	Referência
LCOM - Falta de coesão nos métodos	LCOM = ((n/2) − (Σ* (n − i))) ÷ P	N: número total de métodos das classes P: é o número de conjuntos de métodos que acessam pelo menos uma variável de instância comum.	Quanto maior a coesão de uma classe, maior a clareza e o foco de seus métodos em uma única responsabilidade, facilitando a compreensão e a manutenção do código. Baixa coesão pode tornar o código menos classe e mais difícil de entender e de manter.	[A1] [A2] [A3] [A4] [A6] [A7] [A10] [A15] [A16] [A19] [A21] [A22] [A23] [A24] [A26] [A28] [A29]
TCC – Coesão de Classe Forte	TCC = NDC / NP	NDC: é o número de conexões diretas entre métodos públicos na classe. NP: número máximo de pares que podem ser formados a partir de um conjunto com N elementos. p Para ser calculado o número máximo de pares possíveis, é NP = [N * (N - 1)] / 2. n: métodos públicos.	Quanto maior a coerência melhor é a qualidade do código e menor é a probabilidade de que uma mudança em um módulo afete outros módulos.	[A4] [A23]
CBO – Acoplamento entre objetos	CBO = Ca + Ce	Ca (Acoplamentos aferentes): acoplamentos aferentes de uma classe que efetua a medida de outras classes que utilizam uma classe específica . Ce (Acoplamento eferente): acoplamento eferente de uma classe que foi utilizado na medida de outras classes que utilizam uma classe específica .	Quanto maior o número de pares, maior a sensibilidade a mudanças em outras partes do projeto e, portanto, a manutenção é mais difícil.	[A1] [A3] [A5] [A6] [A7] [A10] [A16] [A17] [A20] [A21] [A22] [A26] [A28] [A29]
DIT – Profundidade da árvore de herança	DIT = Comprimento máximo do caminho da classe até a raiz da árvore de herança Exemplo: P(P + Q) = p(P) = p(Q))	P e Q são as subclasses	Quanto mais profunda uma classe estiver na hierarquia, maior será o número de métodos que será herdado, tornando-a mais complexa prevendo seu comportamento.	[A1] [A2] [A6] [A7] [A10] [A19] [A21] [A22] [A26] [A27] [A28] [A29]
RFC – Resposta para uma classe	RS = {MI U all i {RI}}	RS: conjunto de todos os métodos em uma classe que podem ter respostas ou serem solicitados. (all i {RI}): conjunto de métodos chamados por todos os métodos da classe. MI: representa um método específico (Método i) dentro de uma classe, representa um único método na classe. R: conjunto de métodos chamados pelo método i M: conjunto de todos os métodos da classe	Quanto maior o número de métodos que uma classe pode responder, maior a complexidade dessa classe.	[A1] [A2] [A3] [A6] [A7] [A10] [A19] [A20] [A21] [A22] [A26] [A28] [A29]
WMC – Métodos ponderados por classe	WMC = ∑Ci (para i = 1 a n)	Ci: representa a complexidade do método da classe; n: é o número total de métodos dentro da classe.	Quanto maior o número de métodos em uma classe, maior é o impacto potencial da herança dos métodos definidos nessa classe para as subclasses.	[A1] [A2] [A3] [A6] [A7] [A15] [A17] [A19] [A21] [A22] [A27] [A28] [A29]
CLOC – Contar linhas de código	Vn = Vn-1 + V0	Vn: número de linhas do código Vn-1: tamanho V0: tamanho inicial do software, primeira versão do projeto	O CLOC é um indicador do tamanho do código que pode ter um impacto positivo no índice de manutenção, fornecendo informações ao longo do tempo, garantindo que o software possa ser mantido de forma eficiente e sustentável.	[5] [A29]
CC – Complexidade Ciclomático	CC = E − N + P	E: é o número de arestas do grafo; N: é o número de nós; P: é o número de componentes conectados.	Quanto menor a complexidade, maior é a facilidade de manutenção e menor é a probabilidade de erros.	[A2] [6] [A8] [A10] [A16] [A17] [A18] [A19]
NOC – Número de Filhos	NOC (P + Q) = np + nQ - d	Np: é o número de subclasses imediatas da classe P; nQ: é o número de subclasses imediatas da classe Q; d: é número de filhos em comum entre P e Q.	Quanto maior o número de filhos, maior o reuso, pois a herança é uma forma de reuso. Quanto maior o número de filhos, maior a probabilidade de abstração indevida da classe pai.	[A1] [A2] [A3] [A6] [A7] [A10] [A19] [A21] [A22] [A26] [A27]

Na análise qualitativa dos artigos selecionados, ficou evidente que diversas análises de qualidade exercem influência direta sobre o Índice de Manutenção de Software Orientado a Objeto (OMI). Conforme listado na Tabela 3, destacam-se, entre os mais relevantes, a complexidade ciclomática, a coesão e o acoplamento entre as classes, assim como a taxa de defeitos e a duplicação de código. Essas conclusões contêm insights sobre a estrutura interna do código, sua manutenibilidade e a propensão a erros em futuras modificações.

Como elas impactam?

As métricas de qualidade desempenham papéis específicos no contexto do Índice de Manutenção de Software Orientado a Objeto (OMI). Por exemplo, uma complexidade ciclomática está intrinsecamente ligada à compreensão e à modificação de trechos de código. À medida que a complexidade aumenta, a manutenção torna-se mais desafiadora. Já a união e a proteção entre classes, afetam a interdependência das partes do sistema, influenciando a facilidade de realizar alterações sem impactar outras áreas do software. A taxa de defeitos e a duplicação de código indicam a qualidade global do código e sua propensão a erros, impactando diretamente a estabilidade e a confiabilidade do software. Em resumo, essas avaliações de qualidade exercem impactos diversos, mas todos relevantes no Índice de Manutenção de Software Orientado a Objeto.

Como elas são calculadas?

Conforme ilustrado na Tabela 3, o cálculo das métricas pode variar conforme a ferramenta utilizada, mas geralmente envolve uma contagem de elementos do código, como classes, métodos, linhas de código, superclasses, subclasses, entre outros. O objetivo é obter um valor numérico que indique a qualidade do código em relação à métrica em questão. Cada métrica possui uma fórmula específica para o cálculo, disponível nos documentos das ferramentas de análise de código-fonte.

Para calcular cada métrica, é necessário aplicar a fórmula específica associada a ela. Essas fórmulas dependem da métrica em questão e das ferramentas utilizadas para a análise de código-fonte. O propósito é obter um valor numérico que represente a qualidade do código em relação à métrica específica. Essas análises não são cruciais apenas para avaliar a qualidade do código ao longo do ciclo de vida do software, mas também para identificar áreas de possível aprimoramento na manutenção do software.

Para mensurar o impacto de cada métrica no Índice de Manutenção do software, foi definida uma escala de impacto que varia em Baixo, Médio, Alto e Sem Impacto. A classificação de cada métrica em uma dessas categorias varia de acordo com a quantidade de artigos que citaram tal métrica como sendo impactante no Índice de Manutenção:

- Sem Impacto. Indica que a métrica tem um impacto insignificante no índice de manutenção. Mudanças ou variações nessa métrica não terão um efeito significativo na manutenção do sistema.
- Baixo Impacto (menos de 5 Artigos). Reflete um impacto relativamente pequeno no índice de manutenção. Um número limitado de artigos sugere que as mudanças podem ser gerenciadas com facilidade.
- Médio Impacto (Entre 5 a 10 Artigos). Indica um impacto moderado no índice de manutenção. Com um número moderado de artigos, variações na métrica começam a ter um efeito prático na manutenção, exigindo alguma atenção.
- Alto Impacto (Acima de 10 Artigos). Indica um impacto significativo no índice de manutenção. Um número significativo de artigos sugere que variações nesta métrica podem ter um impacto, exigindo atenção e esforço consideráveis para gerenciar e manter o sistema.

Considerando essa escala, na Tabela 4, são apresentados os resultados obtidos após análise de impacto de cada métrica no Índice de Manutenção.

Tabela 4. Resultados obtidos

	COESÃO	ACOPLAMENTO	TAMANHO DO CÓDIGO	COMPLEXIDADE	ABSTRAÇÃO
LCOM	◉	○	○	○	○
TCC	○	○	○	○	○
NOC	◉	○	◐	◐	●
CC	○	◐	◐	●	●
CLOC	○	○	○	○	●
WMC	◉	◐	◉	◐	○
RFC	◐	◉	◐	◉	◐
DIT	◐	◉	◉	●	○
CBO	◉	◉	◉	◉	◉

Sem Impacto: ● Baixo Impacto: ○ Médio Impacto: ◐ Alto Impacto: ◉

CONCLUSÃO

Este artigo apresentou os resultados de uma Revisão Sistemática da Literatura (RSL) sobre métricas de qualidade de software e seu impacto no Índice de Manutenção em projetos orientados a objetos. Após a seleção primária, 29 artigos foram identificados como relevantes, permitindo uma análise das principais métricas, como coesão, acoplamento, tamanho do código, complexidade, abstração, reusabilidade e testabilidade. Essas métricas foram identificadas como essenciais para compreender a manutenibilidade do software e sua relação com o Índice de Manutenção.

A análise combinada de dados quantitativos e qualitativos validou a importância de métricas como LCOM, TCC, NOC, CC, CLOC, WMC, RFC, DIT e CBO, destacando os desafios na gestão da qualidade de software. Este entendimento fornece diretrizes úteis para profissionais e pesquisadores, fornecendo insumos para análise da qualidade de software e orientação para desenvolvimento e manutenção de melhor qualidade.

O estudo também identifica a necessidade de avanços futuros, especialmente no desenvolvimento de tecnologias que automatizem o cálculo dessas métricas e as integrem em ambientes de desenvolvimento. Essas ferramentas podem facilitar a aplicação das métricas e permitir uma gestão mais eficiente da qualidade do software, contribuindo para o sucesso em projetos orientados a objetos. Assim, a pesquisa oferece dados relavantes capazes de direcionar novas investigações e desenvolvimentos na área.

REFERÊNCIAS

[1] DE SOUZA, Priscila Pereira et al. A utilidade dos valores referência de métricas na avaliação da qualidade de softwares orientados por objeto. 2016.

[2] FILÓ, Tarcisio Guerra Savino. Identificação de valores referência para métricas de softwares orientados por objetos. 2014.

[3] GAIA, Josiane Rosa de Oliveira. Manutenção de software e estudo de caso aplicado ao software Openbiblio. 2013.

[4] WILDE, Michelle. Biblioteca digital Ieee xplore. The Charleston Advisor , v. 17, n. 4, pág. 24-30, 2016.

[5] BAER, Nikolaus; ZEIDMAN, Roberto. Medindo a evolução do software com a mudança de linhas de código. Em: CATA . 2009. pág. 264-170.

[6] JURECZKO, Marian. Significado de diferentes métricas de software na previsão de defeitos. Engenharia de Software: An International Journal , v. 1, n. 1, pág. 86-95, 2011.

[7] MAXIM, Bruce R.; PRESSMAN, Roger S. Software Engineering: A Practitioner'S Approach. Britania Raya: McGraw-Hill Education, 2014.

[8] CHIDAMBER, Shyam R.; KEMERER, Chris F. Um conjunto de métricas para design orientado a objetos. Transações IEEE sobre engenharia de software , v. 6, pág. 476-493, 1994.

[9] FENTON, Norman E.; PFLEEGER, Shari Lawrence. Métricas de software: uma abordagem rigorosa e prática: Brooks. 1998.

[10] BASILI, Victor R.; BRIAND, Lionel C..; MELO, Walcélio L. Validação de métricas de design orientado a objetos como indicadores de qualidade. Transações IEEE sobre engenharia de software , v. 10, pág. 751-761, 1996.

[11] KANAKI, Kalliopi et al. Investigando a Associação entre Pensamento Algorítmico e Desempenho em Estudo Ambiental. Sustentabilidade , v. 14, n. 17, pág. 10672, 2022.

[12] PRESSMAN, Roger S.; MAXIM, Bruce R. Engenharia de software-9. McGraw Hill Brasil, 2021.