A METRICS-BASED APPROACH TO ASSESSING THE QUALITY OF SOFTWARE PROJECTS

UMA ABORDAGEM BASEADA EM MÉTRICAS PARA AVALIAÇÃO DA QUALIDADE DE PROJETOS DE SOFTWARE

SILVA, Rodrigo Alves de Brito Brilhante; RAMOS FILHO, Sebastião Marcos Mayor; CARVALHO, Marcos Alberto; CARVALHO, Jaqueline Corrêa Silva; SANTOS, Flávia Aparecida Oliveira; RAMOS, Celso de Ávila; SILVA, Vinícius Duarte Esteves; SOUZA, Patrícia Carolina; BASTOS, Camila

Rodrigo Alves de Brito Brilhante Silva, UNIFENAS, Brasil

Sebastião Marcos Mayor Ramos Filho, UNIFENAS, Brasil

Marcos Alberto Carvalho, UNIFENAS, Brasil

Jaqueline Corrêa Silva Carvalho, UNIFENAS, Brasil

Flávia Aparecida Oliveira Santos, UNIFENAS, Brasil

Celso de Ávila Ramos, UNIFENAS, Brasil

Vinícius Duarte Esteves Silva, UNIFENAS, Brasil

Patrícia Carolina Souza, UNIFENAS, Brasil

Camila Bastos, 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: 10.29327/2385054.6.5-9

ABSTRACT: Software quality has become prominently highlighted as technology has spread widely across different fields. Quality remains an abstract concept, which is difficult to identify in presence but easily noticed in absence. By analyzing internal quality attributes, it is possible to identify maintainability issues and avoid errors during software use. Thus, a methodology was proposed to measure software coupling and compare the results with reference values. The methodology was implemented in the JMAP tool, a plug-in for the Eclipse IDE that allows for measuring the coupling of Java projects located in the Eclipse workspace. Tests conducted showed that JMAP provided accurate results for coupling metrics in both pilot and real projects. The tool proved effective in analyzing large-scale software, demonstrating good performance in terms of both time and accuracy. The conclusion is that JMAP can be integrated into the development process to improve internal software quality and facilitate the identification of issues before deployment. Future improvements will include expanding the calculated metrics and developing new visualization methods for the results.

KEYWORDS: Software Maintenance; Software Metrics; Java Plug-in.

RESUMO: A qualidade de software tem se destacado de forma expressiva, visto que a tecnologia teve grande propagação em diferentes áreas. A qualidade ainda é um conceito abstrato, cuja presença é difícil de identificar, porém, sua ausência é facilmente percebida. Com a análise dos atributos internos de qualidade, é possível identificar problemas de manutenibilidade e evitar erros durante a utilização do software. Deste modo, foi proposta uma metodologia para mensurar o acoplamento do software e comparar os resultados com valores de referência. A metodologia foi implementada na ferramenta JMAP, um plug-in para a IDE Eclipse que possibilita aferir o acoplamento de projetos Java situados na área de trabalho do Eclipse. Os testes realizados mostraram que a JMAP apresentou resultados precisos para as métricas de acoplamento em projetos piloto e reais. A ferramenta provou ser eficaz na análise de software de grande porte, com um bom desempenho em termos de tempo e precisão. A

conclusão é que a JMAP pode ser integrada ao processo de desenvolvimento para melhorar a qualidade interna do software e facilitar a identificação de problemas antes da implantação. Futuras melhorias incluirão a ampliação das métricas calculadas e o desenvolvimento de novas formas de visualização dos resultados.

PALAVRAS-CHAVE: Manutenção de Software; Métricas de Software; Plug-in Java.

1 INTRODUÇÃO

A qualidade de software tem se destacado de forma expressiva, visto que a tecnologia teve grande propagação em diferentes áreas e está envolvida em diversas atividades. A qualidade do software é um conceito abstrato, cuja presença pode ser difícil de ser notada, porém, sua ausência é facilmente percebida. Um exemplo de problema de qualidade é a lentidão de um sistema, que pode ser ocasionada por uma implementação mal planejada, erros de projeto ou arquitetura inadequada. Decisões tomadas durante a etapa de desenvolvimento podem impactar na avaliação dos atributos de qualidade internos e externos do software [1].

A análise dos atributos internos de qualidade possibilita a avaliação do estado do software em relação às características técnicas que podem gerar custos e trabalhos desnecessários para a equipe de desenvolvimento. Por exemplo, a análise do acoplamento e da coesão possibilitam avaliar a manutenibilidade e a legibilidade do software, viabilizando a tomada de decisões em relação à refatoração de código antes mesmo de colocar o software em produção. Em concordância com [2], cerca de 80% do custo de desenvolvimento de software estão relacionados com a manutenção.

Estudos consideram que, quanto maior for a quantidade de métodos em uma determinada classe, maior é a probabilidade de aumentar o seu acoplamento [3]. Com isso a manutenibilidade do software tende a ser mais complicada de acordo com a sua evolução. O acoplamento pode ser visto de forma ambígua na programação orientada a objetos, visto que esse paradigma de desenvolvimento se concentra na abstração dos objetos e suas relações [4]. Apesar de necessárias, as relações entre objetos podem causar um forte acoplamento, causando um efeito de “onda de água”, o que significa que as mudanças realizadas em um objeto podem impactar negativamente em outros objetos, prejudicando os demais fluxos do sistema.

Diferentes trabalhos foram realizados para possibilitar a mensuração e o controle do acoplamento em software orientado a objetos. Em um estudo [5], foi proposta uma metodologia para aferir o acoplamento entre as classes de um software. A ferramenta proposta realiza análise estática de código-fonte Java para convertê-lo em unidades de compilação e quantificar valores de acoplamento. Outros autores [6], desenvolveram uma ferramenta web para analisar o acoplamento de software Java e Android utilizando sete métricas diferentes.

Apesar de automatizar o cálculo do acoplamento em software Java, as ferramentas mencionadas não orientam o usuário em relação aos valores de referência ideais para o acoplamento. Além disso, são ferramentas externas não integradas à IDE de desenvolvimento e não possuem técnicas de visualização diferenciadas para apresentação dos resultados, podendo dificultar a interpretação dos usuários. Com base nesses fatores, o objetivo deste trabalho é propor uma metodologia para mensurar o acoplamento de sistemas de software orientado a objetos e comparar os resultados obtidos com valores de referência disponíveis da literatura.

2 METODOLOGIA

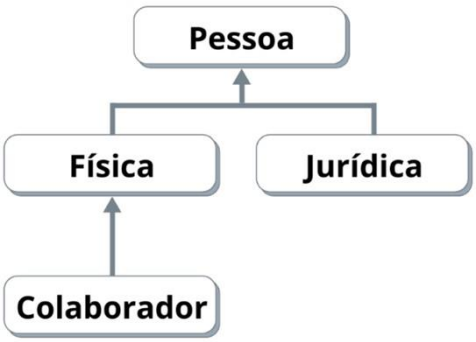
Este trabalho foi desenvolvido seguindo três etapas. Foi realizada uma Análise Bibliográfica, com objetivo de verificar o estado da arte em relação aos assuntos abordados no trabalho. Foram realizadas pesquisas nos principais repositórios de trabalhos científicos utilizando palavras-chave relacionadas ao tema abordado. Dentre os artigos encontrados, foram selecionados aqueles que tratam da melhoria da qualidade interna do software e que abordam o controle do acoplamento e da coesão por meio de métricas.

A etapa de Construção foi executada para definir uma metodologia para mensurar valores de acoplamento em software orientado a objetos e desenvolver uma ferramenta para executar a metodologia. Em relação às métricas de acoplamento, foram selecionadas aquelas mais citadas nos trabalhos científicos, cuja contribuição para a qualidade de software foi cientificamente comprovada nos estudos. Um plugin para operar de forma integrada à IDE Eclipse foi desenvolvido, de modo a automatizar o cálculo das métricas de acoplamento e apresentar os resultados ao usuário durante toda a etapa de desenvolvimento do software.

A etapa de Avaliação foi executada para verificar a execução e a aplicabilidade da metodologia e da ferramenta desenvolvida em cenários reais. A ferramenta foi utilizada para analisar o acoplamento do código-fonte de sistemas de software Java open-source. Os resultados obtidos foram comparados com resultados publicados por trabalhos científicos e analisados.

As métricas utilizadas no desenvolvimento do trabalho foram ocorrência de acoplamento de herança e de abstração de dados. A ocorrência de acoplamento de herança [5][6] ocorre quando há herança entre classes, onde as subclasses podem acessar membros públicos da classe base. Esse relacionamento é considerado um acoplamento forte, visto que mudanças realizadas na classe base podem afetar toda uma hierarquia de subclasses. Na linguagem Java, essa dependência pode ser identificada quando se utiliza os operadores extends ou implements para herança e interfaces, respectivamente. Como ilustrado na Figura 1, a classe Pessoa possui duas ocorrências de acoplamento por herança e a classe Física possui apenas uma ocorrência de acoplamento por herança.

Figura 1. Representação herança de classe



O acoplamento por abstração de dados acontece quando uma classe é utilizada para definir tipo de dados em outra classe ou quando uma classe é utilizada como tipo na passagem de parâmetro ou retorno de um método de outra classe [5][7]. Como ilustrado na Figura 2, há uma Classe A e uma Classe B e existem métodos na Classe A cujo retorno são objetos do tipo Classe B. Além disso, existem métodos da Classe A que possuem como parâmetros objetos do tipo da Classe B.

Figura 2. Representação de abstração de dados

```
public class CLASSE_A
{
    //Retorna um objeto de uma outra classe (Classe B)
    public CLASSE_B getObject()
    {
        CLASSE_B objeto = new CLASSE_B();
        return objeto;
    }

    //Passa um objeto do tipo de outra classe (Classe B) no parametro do método
    public void setObject(CLASSE_B objectParameter)
    {
        CLASSE_B objectB = new CLASSE_B();
        objectB = objectParameter;
    }
}
```

Para estabelecer o estado do acomplamento do software, foram definidos três critérios de classificação: Bom, Regular e Ruim. Estes critérios foram estabelecidos com base em valores de referência recomendados na literatura especializada, que indicam os níveis ideais de acoplamento para garantir uma boa qualidade de software. A classificação "Bom" é atribuída quando os valores obtidos estão abaixo do limite ideal, sugerindo um baixo nível de acoplamento e um design de software eficaz. A classificação "Regular" é aplicada quando os valores estão próximos, mas ainda dentro dos limites aceitáveis, indicando áreas que podem ser aprimoradas. Por fim, a classificação "Ruim" é usada quando os valores excedem os limites recomendados, indicando problemas potenciais com a manutenção e a flexibilidade do software.

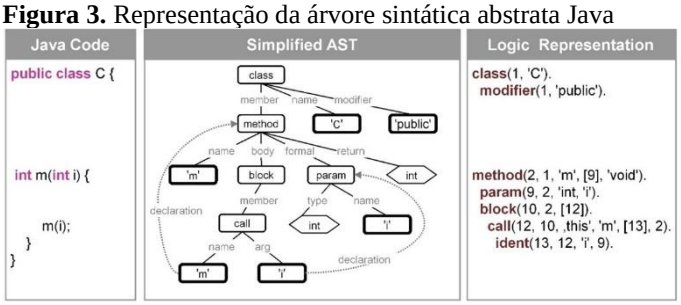
3 RESULTADOS E DISCUSSÃO

3.1 Características da ferramenta desenvolvida

Após a definição das métricas de acoplamento a serem aplicadas, foi desenvolvida uma ferramenta para automatizar o cálculo dessas informações em um software orientado a objetos. Para facilitar a utilização da ferramenta durante a fase de desenvolvimento, optou-se por desenvolver um recurso integrado à IDE de desenvolvimento, permitindo que os desenvolvedores possam aferir a qualidade do software enquanto o codificam. Dessa forma, foi criado o JMAP (Java Metrics Analyzer Plugin), um plugin para a IDE Eclipse desenvolvido utilizando o PDE (Plug-in Development Environment) do Eclipse [8]. Esse recurso possui uma perspectiva que fornece um conjunto de ferramentas para criar, desenvolver, testar e implementar plug-ins para a IDE Eclipse.

As métricas foram calculadas por meio da análise da estrutura do código-fonte dos projetos Java utilizando o JDT (Java Development Tools). O JDT (Java Development Tools) fornece um conjunto de APIs (Application Programming Interface) para acessar e manipular o código-fonte Java, transformando-o em uma AST (Abstract Syntax Tree). Conforme exemplificado na Figura 3, a árvore abstrata possibilita mapear todo o código-fonte do projeto Java em estrutura hierárquica composta por elementos orientados à objetos, permitindo o cálculo das métricas.

Figura 3. Representação da árvore sintática abstrata Java

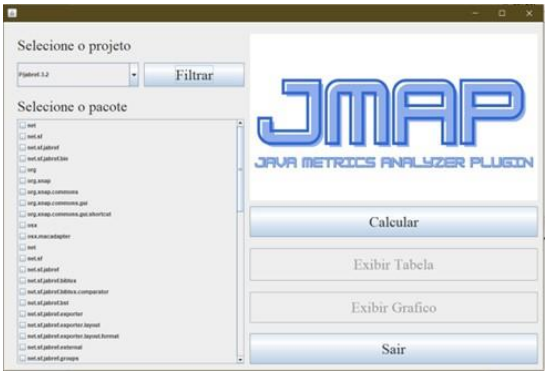


Para melhor visualização e compreensão dos dados obtidos, os valores calculados para as métricas foram apresentados em Jframes da IDE Eclipse, utilizando tabelas e gráficos. Para cada classe presente no modelo de domínio do projeto analisado, são apresentados os valores calculados para as métricas de Acoplamento de Herança e Acoplamento de Abstração de Dados, juntamente com os valores de referência ideais propostos por [9]. O plug-in JMAP pode ser executado por meio de uma combinação de teclas (Crtl + J) ou pela barra de ferramentas da IDE Eclipse, conforme ilustrado na Figura 4.

Figura 4. Meios de acionamento da ferramenta através da IDE Eclipse.

Após iniciar a execução da ferramenta, é exibida uma tela (Jframe) contendo uma dropdown list com todos os projetos disponíveis no workspace, conforme ilustrado na Figura 5. Ao selecionar o projeto desejado, é necessário selecionar também quais pacotes deste projeto que deverão ser analisados para cálculo das métricas. Após a seleção dos itens, basta clicar no botão “Calcular” para que o plug-in inicie o cálculo das métricas propostas.

Figura 5. Janela de exibição principal da ferramenta.



Na Figura 6, é apresentada a visualização da tabela gerada pelo plug-in. A tabela possui cinco colunas, sendo que a primeira coluna apresenta o nome de todas as classes analisadas. A segunda coluna apresenta a contabilização do acoplamento de herança das classes analisadas. A terceira coluna informa se o acoplamento de herança está dentro do padrão de aceitação, podendo ser classificado como: Bom, Regular ou Ruim. A quarta coluna apresenta a contabilização do acoplamento de abstração de dados das classes analisadas. Por fim, a quinta coluna. Informa se o acoplamento de abstração de dados está dentro do padrão de aceitação, podendo ser classificado como: Bom, Regular ou Ruim.

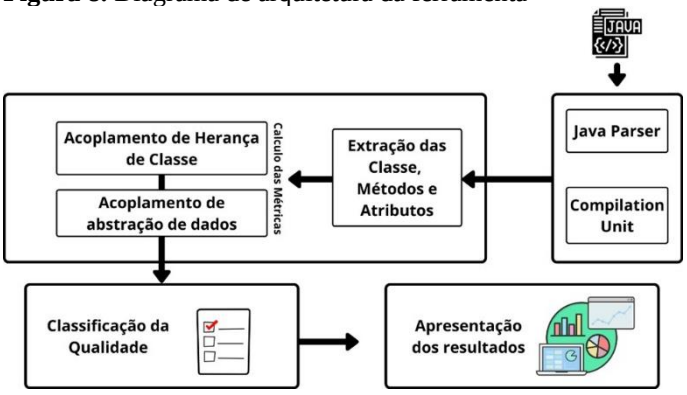
Figura 6. Tabela de resultados da ferramenta

Na visualização em gráfico, apresentada na Figura 7, diferentemente da visualização em tabela, foram definidos filtros que permitem exibir os resultados de apenas uma métrica por vez, separando o Acoplamento de Abstração de Dados do Acoplamento de Herança de classe.

Figura 7. Gráficos de resultados da ferramenta

Na Figura 8 é apresentado o diagrama que ilustra o funcionamento da ferramenta, demostrando as etapas do processo de análise e cálculos das métricas. O código-fonte de um software Java é tratado pela biblioteca JavaParser, que fornece uma AST (Arvore Sintática Abstrata) do código Java. A estrutura AST permite manipular o código-fonte Java, de modo a acessar as unidades de compilação das suas classes, métodos e atributos. Após ter acesso a essas informações, é possível efetuar os cálculos das métricas. Após os cálculos, é realizada a etapa de classificação da qualidade interna.

Figura 8. Diagrama de arquitetura da ferramenta



Para o processo de classificação da qualidade interna, foram utilizados valores de referência sugeridos no trabalho de [9], onde foi realizado um estudo para definir valores ideais para vários tipos de métricas de acoplamento. Nesse trabalho, a métrica de Acoplamento de Herança de Classe (AHC) foi denominada como “Número de Filhas” e a métrica Acoplamento de Abstração de Dados (AAD) foi denominada como “Acoplamento Eferente”, cujos valores de referência foram estabelecidos conforme ilustrado na Tabela 1.

Tabela 1. Catálogo de Valores Referência

Métrica	Bom	Regular	Ruim
AAD	AAD ≤ 6	6 < AAD ≤ 16	AAD > 16
AHC	AHC ≤ 1	1 < AHC ≤ 3	AHC > 3

3.2 Avaliação

A ferramenta foi submetida à análise do código-fonte de diferentes projetos Java. Em seguida, foram analisados a exatidão dos resultados apresentados pela ferramenta e o tempo gasto em cada operação.

3.2.1 Caracterização dos sistemas de software selecionados

Para verificar o funcionamento da ferramenta, foram realizados testes em duas etapas. Na primeira etapa, a ferramenta foi submetida à análise de projetos piloto de pequeno porte. O objetivo dessa etapa é identificar se o cálculo das métricas está sendo realizado corretamente pela ferramenta JMAP. Na segunda etapa, o objetivo é verificar o funcionamento da ferramenta ao ser submetida à análise de projetos reais. Para isso, foram selecionados cinco projetos Java Open source, cujo código-fonte esteja disponível em repositórios na Internet. Os projetos reais selecionados possuem uma quantidade significativa de pacotes, classes e métodos.

3.2.2 Execução da avaliação

Esta etapa teve como principal objetivo verificar a execução e a aplicabilidade da metodologia e da ferramenta proposta em cenários reais. Para serem analisados pela JMAP, é necessário que o código-fonte dos projetos estejam importados no workspace do Eclipse. Primeiramente, foram analisados os projetos piloto. O objetivo foi verificar se a ferramenta está calculando corretamente os valores das métricas. Para isso, foi realizado previamente o cálculo de cada métrica para esses projetos e, em seguida, os valores calculados foram comparados com os resultados obtidos pela ferramenta. O tempo de execução para cada projeto foi coletado utilizando o System.currentTimeMillis, um recurso nativo da linguagem Java que retorna a hora atual do sistema medida em milissegundos.

3.2.3 Análise dos resultados

Na primeira etapa executada para avaliar o funcionamento da ferramenta, foram analisados os projetos piloto. Analisando o diagrama de classes do projeto piloto, foi possível calcular manualmente os valores das métricas de acoplamento de abstração de dados e acoplamento de herança de classe para que, posteriormente, pudessem ser comparados com os resultados obtidos pela ferramenta.

Na Figura 9 foram apresentados os resultados obtidos pela ferramenta após a análise do "Projeto 2", apresentando os valores das métricas para cada classe juntamente com a classificação de qualidade. Na Figura 10, foram apresentados os resultados utilizando o gráfico de barras. Os valores de acoplamento obtidos para esse projeto estão de acordo com o diagrama de classes. Os resultados obtidos com a análise do "Projeto 1" também estão de acordo com o diagrama de classes, mostrando o correto funcionamento da ferramenta em relação aos cálculos das métricas.

Figura 9. Resultados do projeto piloto

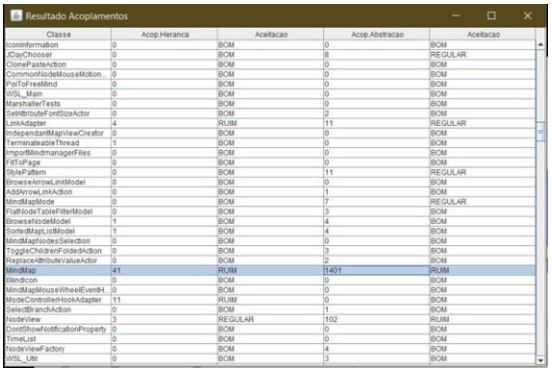
Figura 10. Gráfico de resultados do projeto piloto

Após a análise dos projetos piloto, foi realizada uma outra etapa de testes, cujo objetivo é verificar o funcionamento da ferramenta ao ser submetida a análise de projetos reais. Em relação ao tempo, ocorreu variação para cada projeto, sendo que quanto maior a dimensão do projeto, maior o tempo gasto para cálculo das métricas. Apesar dessa variação, foi possível observar que a ferramenta foi capaz de apresentar os resultados em tempo satisfatório, visto que demandou menos de 1 minuto para analisar um sistema de aproximadamente 3.000 classes e 19.000 métodos, como é o caso do ArgoUML.

Com a análise dos resultados, foi possível observar que, dentre todos os projetos reais analisados, apenas algumas classes desses projetos obtiveram o acoplamento classificado como "Ruim". Na Figura 11, foi apresentada parte dos resultados obtidos após a análise do projeto FreeMind. Todas as classes dos projetos JabRef e JEdit se encontram dentro dos valores de referência. Por outro lado, os sistemas ArgoUML e Freemind possuem algumas classes com

acoplamento “Regular” e “Ruim”. Essas classificações indicaram que a qualidade dessas classes pode estar comprometida e que mudanças devem ser realizadas com cautela, visto que podem afetar vários módulos do software que estão acoplados a elas.

Figura 11. Resultado da análise negativa do FreeMind



Com base nos dados obtidos, foi possível observar que os projetos analisados possuem um valor de acoplamento satisfatório, pois obtiveram uma média superior a 80% de suas classes com acoplamento considerado “Bom”.

CONCLUSÃO

Este trabalho apresenta uma metodologia para mensurar o acoplamento em sistemas de software orientado a objetos, com classificação dos resultados baseada em valores de referência. O plug-in JMAP para a IDE Eclipse, desenvolvido para automatizar o cálculo do acoplamento em projetos Java, demonstrou eficácia ao fornecer resultados claros e visualmente acessíveis. A ferramenta foi testada em projetos piloto e reais, revelando a precisão dos cálculos e a conformidade com os critérios de classificação estabelecidos.

A análise dos resultados indica que a JMAP é eficaz na avaliação do acoplamento e contribui para a manutenção e melhoria da qualidade interna do software. A ferramenta facilita a detecção de problemas de acoplamento e apoia a tomada de decisões durante o desenvolvimento, proporcionando uma abordagem prática e útil para a gestão da qualidade de software.

REFERÊNCIAS

[1] International Organization for Standarization. ISO/IEC Standard 9126: Software Engineering -- Product Quality, part 1. 2001.

[2] G. Szoke, G. Antal, C. Nagy, R. Ferenc, T. Gyimóthy, “Empirical study on refactoring large-scale industrial systems and its effects on maintainability”, The Journal of Systems and Software, 2016, doi: 10.1016/j.jss.2016.08.071.

[3] A. Santos, A. Cunha, N. Macedo and C. Lourenço, "A framework for quality assessment of ROS repositories," 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, 2016, pp. 4491-4496, doi: 10.1109/IROS.2016.7759661.

[4] L. Li, S. Li, Q. Xun, J. Dong, J. Lin, “Method of Coupling Metrics for Object-Oriented Software System Based on CSBG Approach”, Mathematical Problems in Engineering, 2020, doi:10.1155/2020/3428604.

[5] V. Bidve, P. Sarasu, “Tool for Measuring Coupling in ObjectOriented Java Software”, International Journal of Engineering and Technology, Vol. 8, No. 2, 2016.

[6] V. Bidve, P. Sarasu, S. Pathan, G. Pakle, “Web Based Tool for Measuring Coupling in Object-Oriented Software Modules”, International Journal of Intelligent Engineering and Systems, Vol.12, No.4, 2019 DOI: 10.22266/ijies2019.0831.19.

[7] Bertini, A. Tatu and D. Keim, "Quality Metrics in High-Dimensional Data Visualization: An Overview and Systematization," in IEEE Transactions on Visualization and Computer Graphics, vol. 17, no. 12, pp. 2203-2212, Dec. 2011, doi: 10.1109/TVCG.2011.229.

[8] PDE. Eclipse Foundation, c2020. Disponível em: <https://www.eclipse.org/pde>. Acesso em: 10 de fev. de 2024.

[9] T. Filó. “Identificação de valores referência para métricas de softwares orientados por objetos. ”, Repositório Institucional Da UFMG. 2014.