Estudo Observacional

Um estudo das características de qualidade de sistema Java

Pesquisa da matéria de Laboratório de Medição e Experimentação de Software

Ana Flávia de Carvalho Santos [PUC Minas | *anafcs.academico@gmail.com*]
Júlia Moreira Nascimento [PUC Minas *moreira01092002@gmail.com*]

Pontifícia Universidade Católica de Minas Gerais - Coração Eucarístico

Resumo.
O desenvolvimento de software open-source envolve a colaboração de diversos desenvolvedores e a evolução contínua dos sistemas, o que torna essencial investigar fatores que possam influenciar sua qualidade interna. Este estudo observacional analisou 904 repositórios Java extraídos do GitHub, com o objetivo de avaliar a relação entre atributos de processo — popularidade, maturidade, atividade e tamanho — e métricas de qualidade estrutural do código (LCOM, DIT e CBO). Os resultados mostraram que popularidade e maturidade não apresentaram relação significativa com as métricas analisadas, enquanto atividade e tamanho se mostraram mais relevantes, especialmente no aumento do acoplamento entre classes (CBO). Esses achados indicam que fatores ligados à evolução e ao crescimento dos sistemas exercem maior impacto sobre a qualidade interna do código do que aspectos externos como visibilidade ou longevidade dos projetos.

Abstract.
Open-source software development involves the collaboration of multiple developers and the continuous evolution of systems, which makes it essential to investigate factors that may influence internal quality. This observational study analyzed 904 Java repositories extracted from GitHub, aiming to assess the relationship between process attributes — popularity, maturity, activity, and size — and structural code quality metrics (LCOM, DIT, and CBO). The results showed that popularity and maturity had no significant relationship with the analyzed metrics, while activity and size proved more relevant, especially in increasing coupling between classes (CBO). These findings suggest that factors related to system evolution and growth have a greater impact on internal code quality than external aspects such as visibility or longevity of the projects.

Palavras-chave:
Engenharia de Software; Open-Source; Qualidade de Código; Métricas de Software; Repositórios Java.

Keywords:
Software Engineering; Open-Source; Code Quality; Software Metrics; Java Repositories.

1 Introdução

O desenvolvimento de software de código aberto é fundamental para a engenharia de software, pois permite a colaboração de desenvolvedores de diversas experiências e origens contribuem para a criação, manutenção e evolução de um mesmo software. Portanto, essa liberdade de contribuições faz com que a qualidade dos sistemas *open-source* seja uma importante pauta a ser explorada, analisando elementos relacionados à longevidade e confiabilidade desses projetos.

Assim, é importante investigar características do processo de desenvolvimento, como popularidade, maturidade, atividade e tamanho dos repositórios, que estão relacionadas aos parâmetros de qualidade. Estudos em larga escala têm mostrado que atributos como popularidade ou tamanho nem sempre são indicadores diretos de qualidade superior: por exemplo, Roehm *et al*. [2018] analisaram quase 7.000 projetos no GitHub e observaram que sistemas maiores ou mais populares não necessariamente apresentam melhor manutenibilidade, evidenciando a necessidade de investigações específicas sobre tais relações. Esse resultado reforça a importância de compreender como fatores de processo podem se refletir em métricas de qualidade, especialmente em contextos colaborativos e de crescimento acelerado.

Diante disso, este estudo tem como objetivo avaliar as características de qualidade de sistemas Java open-source, analisando em que medida os aspectos do processo de desenvolvimento estão associados às métricas de qualidade interna do código. Para tanto, são propostas quatro questões de pesquisa: (i) verificar a relação entre a popularidade dos repositórios e suas características de qualidade; (ii) analisar se a maturidade influencia essas características; (iii) investigar se a atividade de lançamento de versões está correlacionada à qualidade interna; e (iv) examinar se o tamanho dos repositórios afeta métricas como *CBO (Coupling Between Objects)*, *DIT (Depth of Inheritance Tree)* e *LCOM (Lack of Cohesion of Methods)*.

Com base nessas questões, busca-se contribuir para a compreensão de como atributos de processo podem influenciar a qualidade de sistemas colaborativos, auxiliando pesquisadores e desenvolvedores na adoção de práticas mais eficazes de engenharia de software.

1.1 Objetivos da Pesquisa

O objetivo geral deste trabalho consiste em avaliar as características de qualidade de sistemas Java open-source, analisando como aspectos do processo de desenvolvimento se relacionam com métricas de qualidade interna do código.

A partir desse objetivo, foram definidos os seguintes objetivos específicos:

- Investigar se a popularidade dos repositórios influencia suas características de qualidade;
- Verificar se a maturidade dos repositórios impacta suas características de qualidade;
- Analisar se a atividade (frequência de releases) dos repositórios está relacionada às características de qualidade;
- Examinar se o tamanho dos repositórios afeta suas características de qualidade.

1.2 Questões de Pesquisa e Hipóteses

Com base nos objetivos estabelecidos, o estudo buscou responder às seguintes questões de pesquisa (RQ):

RQ01: Qual a relação entre a popularidade dos repositórios e suas características de qualidade?

- H_0: Não existe relação significativa entre a popularidade (número de estrelas) e as métricas de qualidade (CBO, DIT, LCOM).
- H_{01}: Existe relação significativa entre popularidade e as métricas de qualidade.

RQ02: Qual a relação entre a maturidade dos repositórios e suas características de qualidade?

- H_0: Não existe relação significativa entre a maturidade (idade em anos) e as métricas de qualidade.
- H_{01}: Existe relação significativa entre maturidade e as métricas de qualidade.

RQ03: Qual a relação entre a atividade dos repositórios e suas características de qualidade?

- H_0: Não existe relação significativa entre a atividade (número de releases) e as métricas de qualidade.
- H_{01}: Existe relação significativa entre atividade e as métricas de qualidade.

RQ04: Qual a relação entre o tamanho dos repositórios e suas características de qualidade?

- H_0: Não existe relação significativa entre o tamanho (LOC e linhas de comentários) e as métricas de qualidade.
- H_{01}: Existe relação significativa entre tamanho e as métricas de qualidade.

2 Metodologia

Este estudo adota uma abordagem quantitativa, baseando-se na coleta e análise de dados numéricos para identificar padrões, relações e correlações entre variáveis. Nesse caso, buscou-se investigar como métricas de processo (popularidade, maturidade, atividade e tamanho) podem se relacionar a métricas de qualidade interna de sistemas Java open-source. O objetivo foi identificar possíveis relações estatísticas entre atributos do processo de desenvolvimento e características estruturais do código.

2.1 Seleção dos Repositórios

Foram selecionados 1.000 repositórios Java mais populares do GitHub, para garantir uma amostra válida de sistemas amplamente utilizados pela comunidade. A escolha da linguagem Java deve-se à sua relevância histórica e acadêmica, além da ampla disponibilidade de ferramentas de análise estática que oferecem suporte a projetos desenvolvidos nessa linguagem.

2.2 Coleta de Dados

O processo de coleta de dados iniciou-se com a seleção dos mil repositórios Java mais populares do GitHub, utilizando a API GraphQL disponibilizada pela plataforma. Desse total, foi possível clonar 904 repositórios, uma vez que alguns apresentaram erros de compatibilidade entre a ferramenta CK e versões mais recentes do Java.

Para este estudo, a popularidade dos repositórios foi medida pelo número de *Stargazers*, isto é, usuários do GitHub que marcaram o repositório com uma "estrela". Esse indicador é amplamente utilizado como métrica de engajamento da comunidade e serve como proxy de popularidade dos projetos na plataforma.

Após a clonagem, aplicou-se a ferramenta CK para extrair as métricas de cada projeto. Esse procedimento gerou, para cada repositório, dois arquivos distintos: um contendo informações em nível de classe e outro em nível de método. As métricas de interesse para este estudo, presentes nos arquivos de classes, foram consolidadas em um conjunto de dados no formato CSV.

Em seguida, realizou-se a fusão entre os dois arquivos CSV gerados, de modo a unificar os dados de cada repositório em um único arquivo. A partir desse dataset consolidado, foram conduzidas as análises estatísticas e a geração de visualizações utilizando bibliotecas como Seaborn, entre outras.

2.3 Métricas de Qualidade

Para avaliar a qualidade interna do código dos repositórios mineirados, foram utilizadas três métricas clássicas da suíte CK proposta por Chidamber and Kemerer [1994], Harrison *et al*. [1998] e Okike [2010]. Essas métricas são amplamente empregadas em pesquisas empíricas sobre design orientado a objetos e estão associadas a atributos como manutenibilidade, compreensibilidade e reutilização.

A métrica *CBO (Coupling Between Objects)* mede o número de classes em que uma determinada classe está acoplada, ou seja, quantas dependências ela estabelece ou recebe de outras classes. Valores elevados de CBO indicam forte interdependência entre módulos, o que pode comprometer a manutenibilidade e aumentar o risco de propagação de defeitos. Em contrapartida, valores reduzidos sugerem classes mais independentes e reutilizáveis.

A métrica *DIT (Depth of Inheritance Tree)* representa a profundidade da árvore de herança, correspondendo ao maior caminho entre uma classe e a raiz de sua hierarquia. Um valor elevado de DIT indica maior reutilização de código por herança, mas também pode aumentar a complexidade do sistema, dificultando o entendimento e a manutenção. Já valores baixos sugerem hierarquias mais rasas, de menor complexidade, mas com menor reaproveitamento por herança.

Por fim, a métrica *LCOM (Lack of Cohesion of*

Methods) avalia a coesão de uma classe, observando o grau de compartilhamento de variáveis de instância entre seus métodos. Classes com baixo LCOM (alto grau de coesão) tendem a apresentar responsabilidades bem definidas, sendo mais fáceis de compreender e manter. Por outro lado, valores elevados indicam baixa coesão, sugerindo que a classe desempenha múltiplas responsabilidades e pode demandar refatoração.

Essas três métricas permitem analisar a qualidade de sistemas orientados a objetos, possibilitando a identificação de fragilidades estruturais e orientando estratégias de melhoria do design.

3 Resultados

Para responder às questões de pesquisa, foram analisadas as métricas *LCOM (Lack of Cohesion of Methods)*, *DIT (Depth of Inheritance Tree)* e *CBO (Coupling Between Objects)* em relação às variáveis de processo investigadas (popularidade, maturidade, atividade e tamanho dos repositórios). Essas métricas refletem diferentes aspectos da qualidade interna do código.

3.1 RQ01: Qual a relação entre a popularidade dos repositórios e suas características de qualidade?

A popularidade dos repositórios foi medida pelo número de Stargazers (usuários que marcaram o repositório com estrela no GitHub) e analisada em relação às métricas de qualidade interna do código (*LCOM*, *DIT* e *CBO*).

3.1.1 LCOM (Lack of Cohesion of Methods)

A análise indicou uma correlação de Spearman ρ = -0,014 (p-valor = 0,677), apontando ausência de relação significativa entre popularidade e coesão das classes. Isso sugere que repositórios mais populares não necessariamente apresentam maior ou menor coesão.

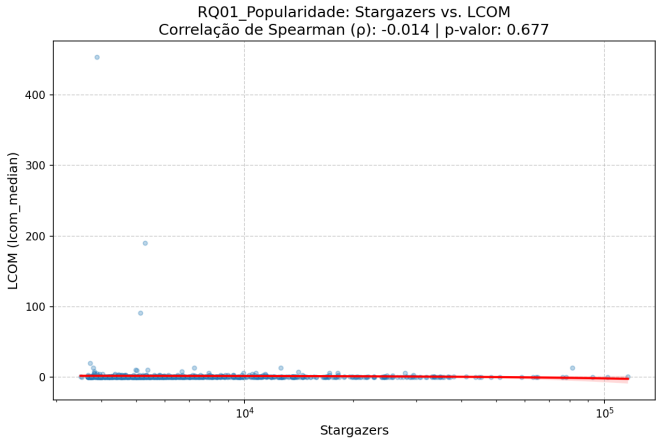

Figura 1. Popularidade Stargazers vs. LCOM

3.1.2 DIT (Depth of Inheritance Tree)

O resultado mostrou um coeficiente de Spearman ρ = -0,040 (p-valor = 0,241), evidenciando falta de correlação significativa entre a popularidade e a profundidade da hierarquia de herança. Assim, a adoção de herança mais profunda ou mais rasa não está associada ao nível de popularidade dos projetos.

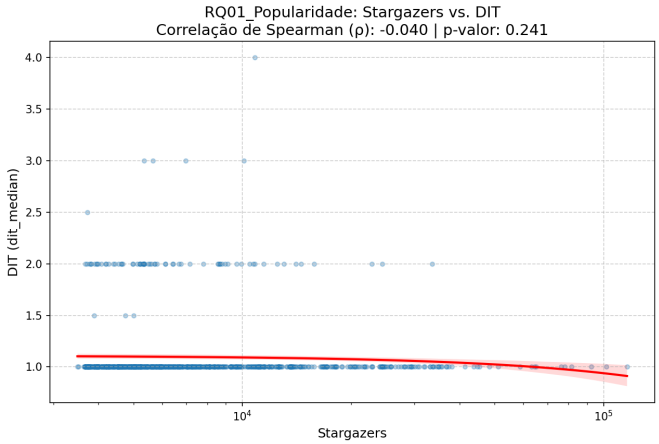

Figura 2. Popularidade Stargazers vs. DIT

3.1.3 CBO (Coupling Between Objects)

A correlação encontrada foi ρ = 0,024 (p-valor = 0,477), também não significativa, mostrando que o acoplamento entre classes não se relaciona diretamente com a popularidade. Projetos populares podem apresentar tanto alto quanto baixo acoplamento sem um padrão claro.

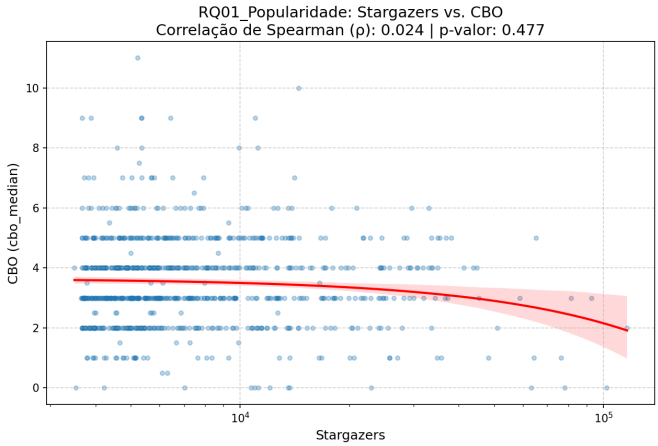

Figura 3. Popularidade Stargazers vs. CBO

3.2 RQ02: Qual a relação entre a maturidade dos repositórios e suas características de qualidade?

A maturidade dos repositórios foi medida em anos desde a sua criação (Age Years) e analisada em relação às métricas de qualidade interna do código (LCOM, DIT e CBO).

3.2.1 LCOM (Lack of Cohesion of Methods)

A análise apresentou uma correlação de Spearman fraca e positiva (ρ = 0,079, p-valor = 0,020). Embora estatisticamente significativa, a força da relação é muito baixa, indicando que a maturidade dos repositórios exerce pouca influência sobre a coesão das classes. Na prática, repositórios mais antigos não apresentam diferenças relevantes em termos de coesão quando comparados aos mais recentes.

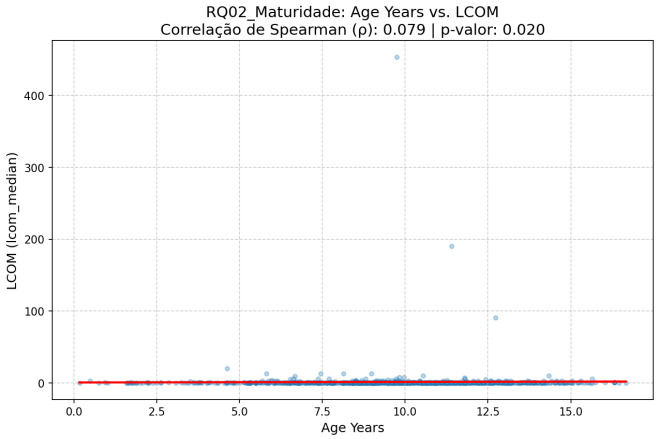

Figura 4. Idade do repositório vs. LCOM

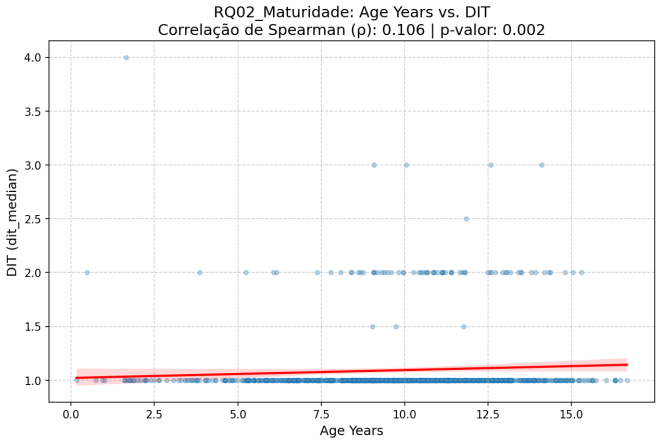

Figura 6. Idade do repositório vs. DIT

3.2.2 DIT (Depth of Inheritance Tree)

O resultado mostrou correlação positiva fraca (ρ = 0,106, p-valor = 0,002). Esse valor sugere que, à medida que os repositórios envelhecem, pode haver uma tendência de estruturas de herança ligeiramente mais profundas. No entanto, o efeito ainda é pequeno, não configurando um padrão forte.

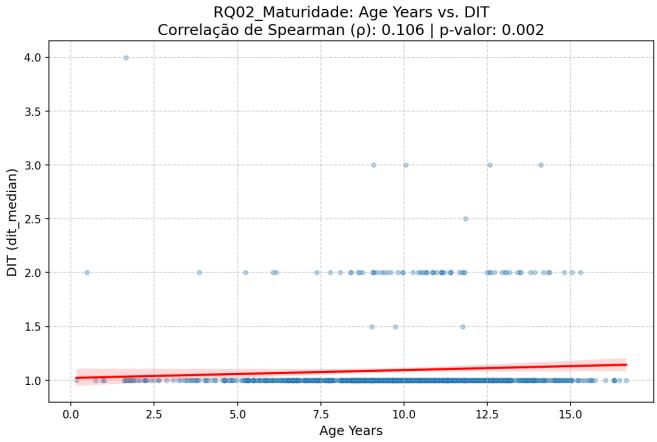

Figura 5. Idade do repositório vs. DIT

3.2.3 CBO (Coupling Between Objects)

Nesse caso, a correlação encontrada foi muito baixa e não significativa (ρ = 0,031, p-valor = 0,368), o que indica ausência de relação entre a maturidade dos repositórios e o acoplamento entre classes. Projetos mais antigos e mais novos podem apresentar tanto alto quanto baixo acoplamento, sem distinção clara.

3.3 RQ03: Qual a relação entre a atividade dos repositórios e suas características de qualidade?

A atividade dos repositórios foi medida pela frequência de releases (lançamentos de versões) e analisada em relação às métricas de qualidade interna do código (LCOM, DIT e CBO).

3.3.1 LCOM (Lack of Cohesion of Methods)

A análise mostrou uma correlação positiva fraca, porém estatisticamente significativa (ρ = 0,102, p-valor = 0,002). Isso indica que repositórios com maior frequência de releases tendem a apresentar leve aumento na falta de coesão das classes. Apesar disso, o efeito é pequeno e não representa uma relação forte.

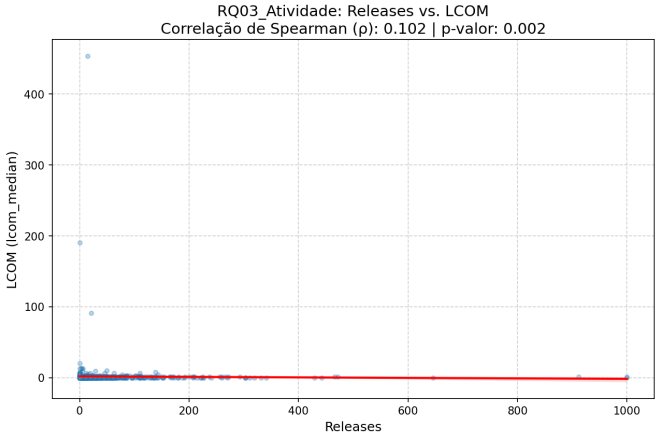

Figura 7. Releases vs. LCOM

3.3.2 DIT (Depth of Inheritance Tree)

O resultado revelou uma correlação negativa muito fraca e não significativa (ρ = -0,033, p-valor = 0,330). Assim, não há evidências de que a quantidade de releases impacte a profundidade das hierarquias de herança.

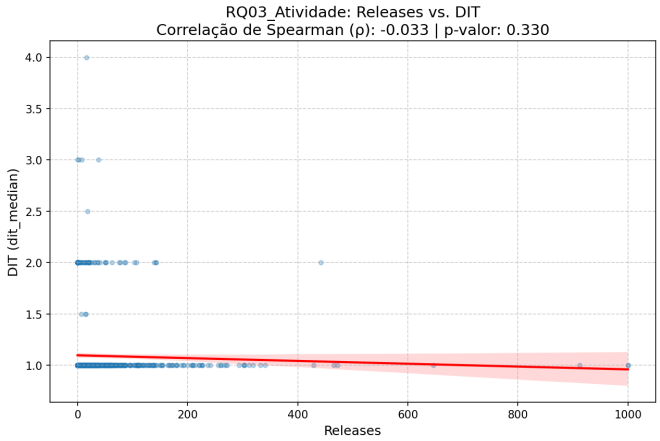

Figura 8. Releases vs. DIT

3.3.3 CBO (Coupling Between Objects)

Nesse caso, observou-se uma correlação positiva moderada e significativa (ρ = 0,280, p-valor = 0,000). Isso sugere que repositórios mais ativos, com maior número de releases, tendem a apresentar maior acoplamento entre classes, possivelmente em função da evolução e complexificação do sistema ao longo do tempo.

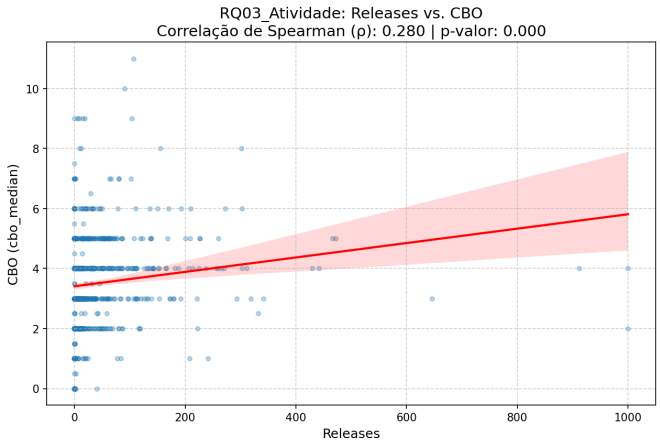

Figura 9. Releases vs. CBO

3.4 RQ04: Qual a relação entre o tamanho dos repositórios e suas características de qualidade?

O tamanho dos repositórios foi medido pelo número de linhas de código (LOC) e linhas de comentários, sendo analisado em relação às métricas de qualidade interna do código (LCOM, DIT e CBO).

3.4.1 LCOM (Lack of Cohesion of Methods)

A análise revelou uma correlação positiva fraca, porém significativa (ρ = 0,141, p-valor = 0,000). Isso indica que repositórios maiores tendem a apresentar classes ligeiramente menos coesas, embora o efeito seja pequeno e não configure uma relação forte.

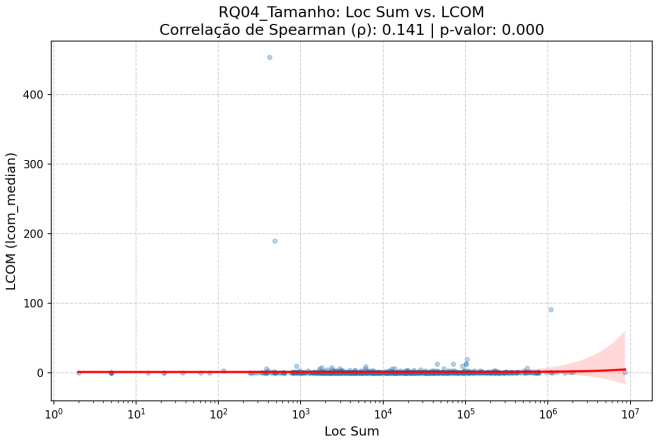

Figura 10. Tamanho do repositório vs. LCOM

3.4.2 DIT (Depth of Inheritance Tree)

Foi observada uma correlação negativa muito fraca e não significativa (ρ = -0,021, p-valor = 0,538). Assim, não há indícios de que o aumento do tamanho do repositório impacte a profundidade da hierarquia de herança.

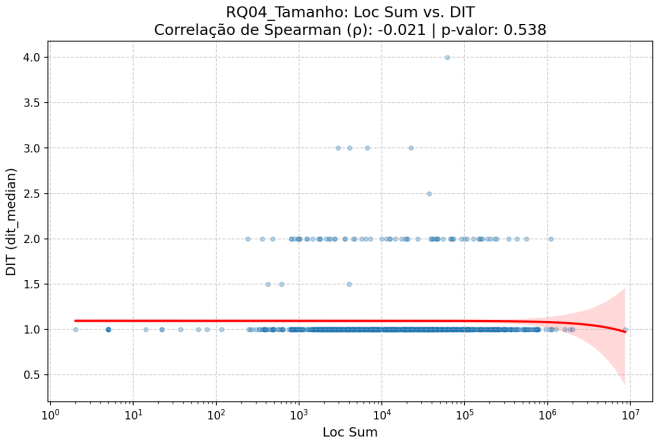

Figura 11. Tamanho do repositório vs. DIT

3.4.3 CBO (Coupling Between Objects)

Neste caso, verificou-se uma correlação positiva moderada e estatisticamente significativa (ρ = 0,293, p-valor = 0,000). Isso sugere que repositórios maiores apresentam maior acoplamento entre classes, o que pode refletir a complexidade crescente do sistema à medida que seu código se expande.

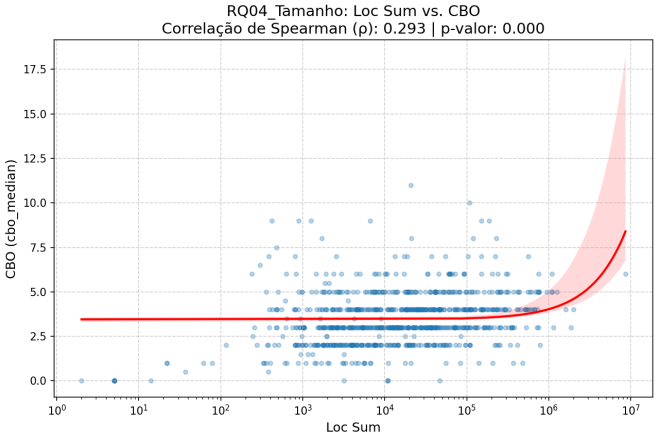

Figura 12. Tamanho do repositório vs. CBO

4 Discussão

Os resultados obtidos permitiram avaliar em que medida as hipóteses nulas (H_0) e alternativas (H_{01}) propostas em cada questão de pesquisa se sustentam. No caso da **RQ01**, que analisava a relação entre a popularidade dos repositórios (medida em número de *Stargazers*) e as métricas de qualidade, a hipótese nula (H_0) afirmava que não existe relação significativa entre essas variáveis, enquanto a alternativa (H_{01}) sugeria que tal relação existe. A análise, entretanto, apontou correlações muito fracas e estatisticamente não significativas para *LCOM*, *DIT* e *CBO*, o que levou à manutenção da hipótese nula e à rejeição da hipótese alternativa.

Na **RQ02**, sobre a maturidade dos repositórios (idade em anos), a hipótese nula (H_0) defendia a ausência de relação significativa com as métricas de qualidade, enquanto a hipótese alternativa (H_{01}) propunha a existência dessa relação. Os resultados revelaram correlações fracas e estatisticamente significativas apenas para *LCOM* e *DIT*, mas sem efeito prático relevante, e ausência de associação para *CBO*. Dessa forma, os achados reforçam majoritariamente a hipótese nula, sugerindo que a idade dos repositórios exerce pouca influência sobre a qualidade interna.

Já a **RQ03**, que avaliou a atividade dos repositórios (número de releases), apresentou um cenário diferente. A hipótese nula (H_0) previa a inexistência de relação, enquanto a alternativa (H_{01}) defendia a existência de associação entre atividade e qualidade. Embora os resultados para *LCOM* e *DIT* tenham sustentado a hipótese nula, o *CBO* apresentou correlação positiva moderada e estatisticamente significativa, o que rejeita parcialmente a hipótese nula e dá lugar à hipótese alternativa. Isso indica que repositórios mais ativos tendem a apresentar maior acoplamento entre classes, possivelmente em razão da complexificação do sistema ao longo do tempo.

Por fim, na **RQ04**, que analisou o tamanho dos repositórios (medido em linhas de código e linhas de comentários), a hipótese nula (H_0) afirmava que não existe relação significativa com as métricas de qualidade, enquanto a hipótese alternativa (H_{01}) sugeria a existência dessa relação. Os resultados mostraram que o tamanho não influencia o *DIT*, confirmando a hipótese nula para essa métrica. Entretanto, para *LCOM* e, principalmente, para *CBO*, foram encontradas correlações positivas e significativas, sendo que no caso do acoplamento a associação foi moderada. Esses achados rejeitam parcialmente a hipótese nula e dão suporte à hipótese alternativa, evidenciando que repositórios maiores tendem a ter menor coesão e maior acoplamento.

Em resumo, foi observado que a popularidade e maturidade não apresentaram influência significativa sobre as métricas de qualidade, sustentando em grande medida as hipóteses nulas, enquanto atividade e tamanho mostraram efeitos mais consistentes, especialmente em relação ao acoplamento entre classes (*CBO*), o que reforça a hipótese alternativa nessas dimensões. Esse padrão sugere que aspectos ligados à evolução e ao crescimento dos sistemas exercem maior impacto na estrutura do código do que fatores externos como visibilidade (popularidade) ou longevidade (maturidade).

5 Conclusão

Este estudo buscou investigar a relação entre atributos de processo (popularidade, maturidade, atividade e tamanho) e métricas de qualidade interna (*LCOM*, *DIT* e *CBO*) em repositórios Java open-source. Os resultados mostraram que popularidade e maturidade não apresentam relação significativa com a qualidade estrutural, sustentando em grande parte suas hipóteses nulas. Em contrapartida, atividade e tamanho revelaram associações mais consistentes, sobretudo no aumento do acoplamento entre classes (*CBO*), sugerindo que fatores ligados à evolução e ao crescimento dos sistemas exercem maior impacto na qualidade do código. Assim, conclui-se que a visibilidade e a longevidade dos projetos não garantem qualidade interna superior, enquanto o volume de código e a intensidade de desenvolvimento podem gerar complexidade adicional que precisa ser gerida pelos desenvolvedores.

Referências

Chidamber, S. and Kemerer, C. (1994). A metrics suite for object-oriented design. *IEEE Transactions on Software Engineering*, 20(6):476–493. DOI: 10.1109/32.295895.

Harrison, R., Counsell, S., and Nithi, R. (1998). Coupling metrics for object-oriented design. In *Proceedings Fifth International Software Metrics Symposium. Metrics (Cat. No. 98TB100262)*, pages 150–157. IEEE.

Okike, E. (2010). A pedagogical evaluation and discussion about the lack of cohesion in method (lcom) metric using field experiment. *arXiv preprint arXiv:1004.3277*.

Roehm, T., Veihelmann, D., Wagner, S., and Juergens, E. (2018). Evaluating maintainability prejudices with a large-scale study of open-source projects. In *International Conference on Software Quality*, pages 151–171. Springer.