Pontifícia Universidade Católica de Minas Gerais

Campus Coração Eucarístico
Engenharia de Software

Trabalho apresentado à disciplina de
Laboratório de Experimentação de Software
Prof. Danilo de Quadros Maia Filho

Contagem – MG

Sumário

1 Introdução

O desenvolvimento de software de código aberto (*open-source*) transformou a maneira como a tecnologia é criada, promovendo um ambiente colaborativo onde desenvolvedores de todo o mundo podem contribuir para um projeto comum. No entanto, essa natureza distribuída e muitas vezes assíncrona impõe desafios significativos à gestão da qualidade interna do código. Atributos essenciais como manutenibilidade, modularidade e coesão podem se deteriorar ao longo do tempo se não houver práticas de engenharia de software bem definidas, como revisões de código rigorosas e análise estática contínua.

Neste contexto, este estudo busca investigar empiricamente a possível correlação entre as características observáveis do processo de desenvolvimento de um projeto e a qualidade intrínseca de seu código-fonte. O objetivo principal é analisar se métricas de processo — como popularidade, idade, atividade e tamanho de um repositório — podem servir como indicadores para métricas de qualidade de produto, como acoplamento (CBO), profundidade de herança (DIT) e coesão (LCOM). Para isso, realizei uma análise em larga escala sobre os 1.000 repositórios da linguagem Java mais populares na plataforma GitHub.

Para guiar esta investigação, o trabalho se propõe a responder às seguintes quatro questões de pesquisa (RQs):

- **RQ 01:** Qual a relação entre a popularidade dos repositórios e as suas características de qualidade?
- **RQ 02:** Qual a relação entre a maturidade dos repositórios e as suas características de qualidade?
- **RQ 03:** Qual a relação entre a atividade dos repositórios e as suas características de qualidade?
- **RQ 04:** Qual a relação entre o tamanho dos repositórios e as suas características de qualidade?

Com base na experiência prática e em percepções comuns da comunidade de desenvolvimento, formulei as seguintes hipóteses iniciais para cada questão, que serão validadas ou refutadas por meio da análise de dados:

- **Hipótese para RQ01:** Acredito que a **popularidade e a qualidade são diretamente proporcionais**. Repositórios mais populares (com mais estrelas) tendem a atrair uma comunidade maior e mais qualificada, o que levaria a um maior escrutínio e a mais contribuições para a melhoria da qualidade do código.
- **Hipótese para RQ02:** Hipotetizo que **repositórios mais maduros apresentam melhor qualidade**. Projetos com mais tempo de existência teriam passado por mais ciclos de refatoração e aperfeiçoamento, resultando em um código mais estável e bem estruturado em comparação com projetos mais novos.
- **Hipótese para RQ03:** Minha suposição é que **repositórios mais ativos possuem uma qualidade inferior**. A alta frequência de releases e mudanças pode indicar um estado de evolução constante, com a introdução de *hotfixes* e novas funcionalidades que podem degradar temporariamente a qualidade. Repositórios menos ativos, por outro lado, seriam considerados mais consolidados e estáveis.

- **Hipótese para RQ04:** Por fim, acredito que **quanto maior o repositório, menor será sua qualidade**. O aumento da base de código tende a elevar a complexidade e o acúmulo de débito técnico, o que se refletiria negativamente nas métricas de qualidade.

2 Metodologia

Para responder às questões de pesquisa propostas, foi executado um processo metodológico em três etapas principais: (1) seleção dos repositórios e coleta de métricas de processo via API do GitHub; (2) extração de métricas de qualidade do código-fonte utilizando a ferramenta CK; e (3) tratamento e análise estatística dos dados coletados. Todo o processo foi automatizado por meio de scripts em Python.

2.1 Seleção dos Repositórios e Coleta de Métricas de Processo

O objeto de estudo deste trabalho foram os repositórios de código aberto da linguagem Java. A primeira etapa consistiu na seleção de uma amostra representativa de projetos relevantes da comunidade. Para isso, foi desenvolvido um script que utiliza a API GraphQL do GitHub para requisitar a lista dos 1.000 repositórios Java públicos com o maior número de estrelas (*stars*).

Para cada repositório da lista obtida, o mesmo script realizou uma segunda consulta à API para extrair as seguintes métricas de processo:

- **Popularidade:** O número total de estrelas (`stargazerCount`).
- **Maturidade:** A idade do repositório em anos, calculada a partir da sua data de criação (`createdAt`).
- **Atividade:** O número total de *releases* (`releases.totalCount`) publicadas no repositório.

Ao final desta etapa, os dados de processo foram salvos em um arquivo CSV para uso posterior.

2.2 Coleta de Métricas de Qualidade de Código

Com a lista de repositórios definida, a etapa seguinte focou na extração de métricas de qualidade de produto. Um script de automação foi utilizado para clonar localmente o código-fonte de cada um dos 1.000 repositórios.

Em seguida, a ferramenta de análise estática de linha de comando **CK** foi executada sobre o código-fonte de cada projeto. O CK analisa o código Java e calcula um conjunto de métricas de software orientadas a objetos, gerando um arquivo de saída em formato `.csv` para cada repositório. As métricas de qualidade selecionadas para este estudo, conforme o escopo do laboratório, foram:

- **CBO (Coupling Between Objects):** Mede o nível de acoplamento de uma classe com outras.

- **DIT (Depth of Inheritance Tree):** Indica a profundidade de uma classe na árvore de herança.
- **LCOM (Lack of Cohesion of Methods):** Mede a falta de coesão dos métodos dentro de uma classe.
- **LOC (Lines of Code):** A contagem de linhas de código, utilizada como métrica de tamanho.

2.3 Tratamento e Análise dos Dados

A saída da ferramenta CK gera métricas para cada classe individualmente. Para possibilitar a comparação entre repositórios, foi necessário agregar esses dados. Primeiramente, um script consolidou todos os arquivos `.csv` individuais em um único conjunto de dados.

Na sequência, o mesmo script processou este conjunto de dados. Para cada repositório, foi calculada a **mediana** de cada métrica de qualidade (CBO, DIT, LCOM) e o **somatório** do LOC. A mediana foi escolhida como medida de tendência central para as métricas de qualidade por sua robustez estatística, pois ela não é significativamente afetada por valores extremos (*outliers*).

Finalmente, o conjunto de dados de processo foi mesclado com o conjunto de dados de qualidade (já agregado), criando uma base de dados final onde cada linha representava um único repositório com suas respectivas métricas. A análise subsequente foi realizada sobre esta base de dados final.

3 Resultados e Análise

Nesta seção, são apresentados os resultados obtidos a partir da análise estatística dos dados coletados dos 1.000 repositórios Java. A análise inicial foca numa caracterização geral do conjunto de dados, seguida por investigações específicas para responder a cada questão de pesquisa. Para mitigar o efeito de valores extremos, o 1

3.1 Análise Geral e Caracterização do Conjunto de Dados

Para compreender a natureza dos dados, a primeira etapa consistiu em analisar a distribuição das métricas de processo e de qualidade. Para complementar a análise visual, a Figura 1 ilustra a distribuição das métricas de processo e tamanho. Observa-se que a maioria dos repositórios tende a concentrar-se na faixa inferior de popularidade, atividade e tamanho (LOC), com uma longa cauda à direita, indicando que poucos projetos atingem valores muito elevados. A maturidade, por outro lado, apresenta uma distribuição mais próxima da normal, com uma concentração de projetos em torno dos 10 a 12 anos de existência.

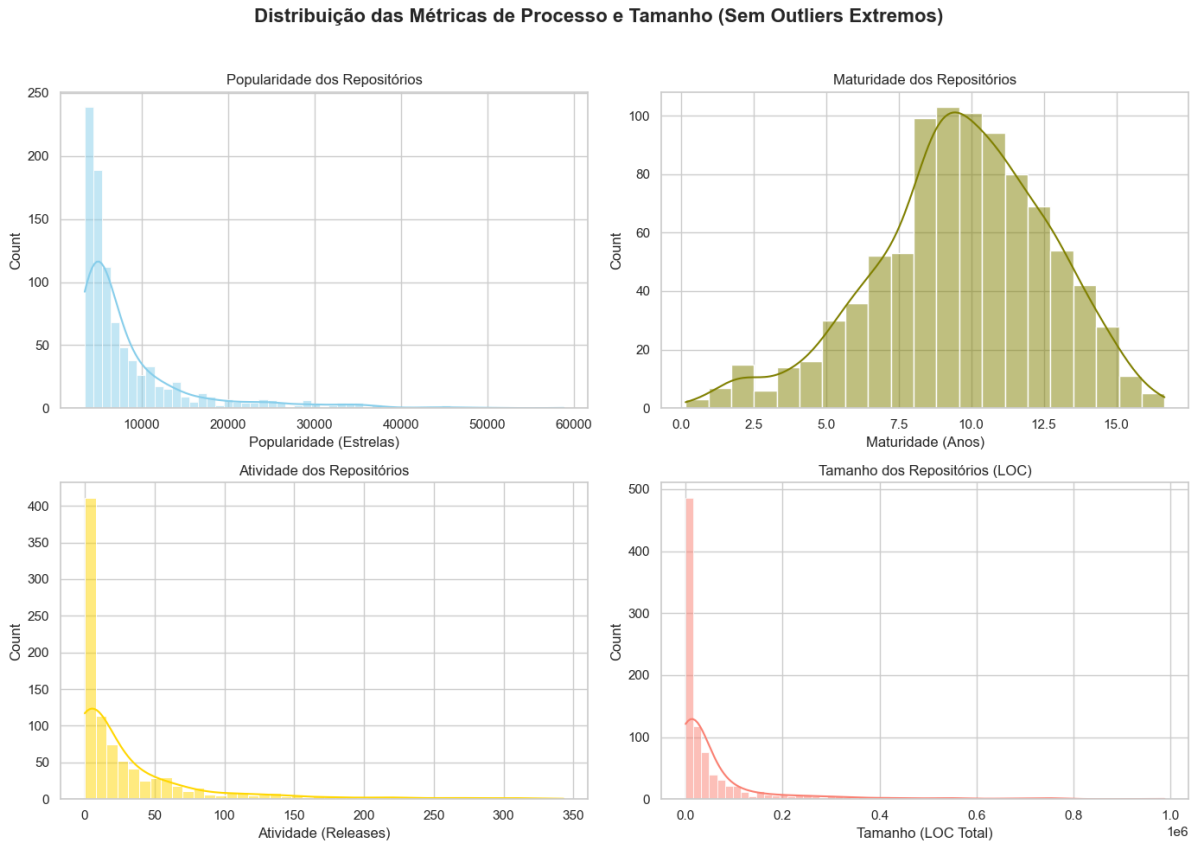

Figura 1: Distribuição das métricas de processo e tamanho.

A análise da dispersão das métricas de qualidade, apresentada na Figura 2, revela que a mediana do DIT (Profundidade de Herança) é muito baixa na maioria dos projetos, concentrando-se em 1, o que sugere um uso limitado de herança profunda. O CBO (Acoplamento) também apresenta valores medianos baixos, geralmente abaixo de 5. O LCOM (Falta de Coesão) mostra uma dispersão ligeiramente maior, mas ainda assim com a maioria dos projetos a apresentar boa coesão (valores baixos).

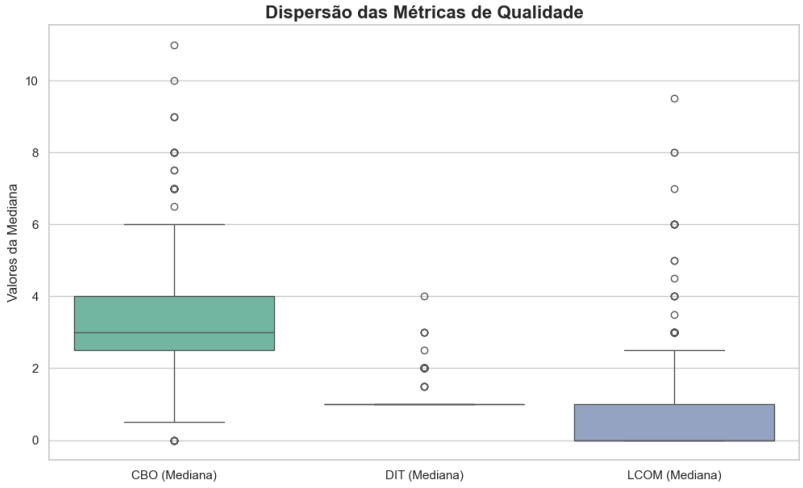

Figura 2: Dispersão das medianas das métricas de qualidade (CBO, DIT e LCOM).

Finalmente, foi gerada uma matriz de correlação de Pearson (Figura 3) para obter uma visão panorâmica das relações lineares entre as variáveis. A análise do mapa de calor indica que, de modo geral, as correlações entre as métricas de processo (popularidade, maturidade, atividade) e as métricas de qualidade (CBO, DIT, LCOM) são muito fracas, com todos os coeficientes entre -0.09 e 0.16. A correlação mais notável, embora ainda fraca, é entre CBO e DIT ($r = 0.29$), o que é esperado, já que estruturas de herança mais profundas podem levar a um maior acoplamento.

Esta visão geral sugere que as relações entre as características do processo de desenvolvimento e a qualidade do código podem não ser lineares ou diretas. As próximas subseções irão investigar estas relações mais a fundo para responder a cada questão de pesquisa.

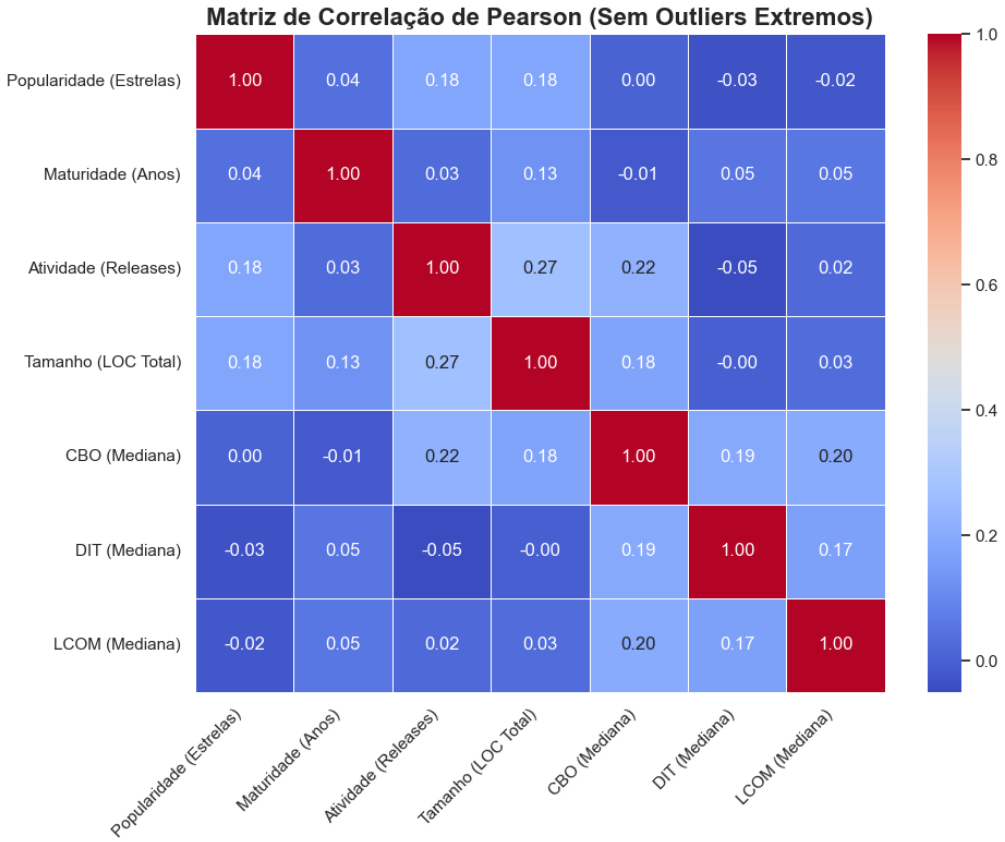

Figura 3: Mapa de calor da correlação de Pearson entre as métricas.

3.2 RQ01: Relação entre Popularidade e Qualidade

A primeira questão de pesquisa buscou investigar se a popularidade de um repositório, medida pelo seu número de estrelas, se correlaciona com as suas características de qualidade. A hipótese inicial (**H1**) era de que a relação seria diretamente proporcional, ou seja, repositórios mais populares tenderiam a apresentar melhor qualidade de código devido a um maior escrutínio da comunidade.

Para validar esta hipótese, foram gerados gráficos de dispersão que comparam a popularidade com as medianas de CBO, DIT e LCOM, conforme apresentado na Figura 4. A análise inclui o cálculo do coeficiente de correlação de Pearson (r) para quantificar a força da relação linear.

A análise dos resultados mostra uma ausência de correlação significativa entre as variáveis:

- A correlação entre Popularidade e **CBO (Acoplamento)** foi de $\mathbf{r = -0.01}$, um valor praticamente nulo.
- A correlação entre Popularidade e **DIT (Herança)** foi de $\mathbf{r = 0.05}$, indicando uma relação linear muito fraca.
- A correlação entre Popularidade e **LCOM (Coesão)** foi de $\mathbf{r = 0.02}$, também um valor insignificante.

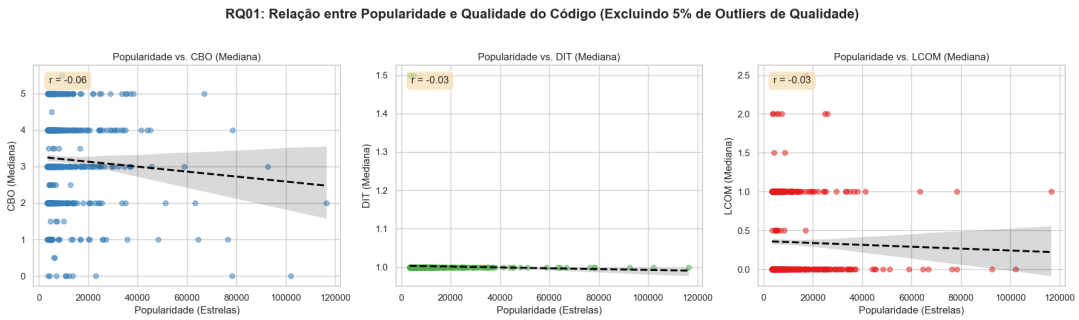

Figura 4: Gráficos de dispersão entre a popularidade (número de estrelas) e as medianas de CBO, DIT e LCOM, com os outliers de qualidade removidos.

Os coeficientes, todos muito próximos de zero, demonstram que não há uma relação linear entre a popularidade de um repositório e as métricas de qualidade interna do seu código. Os gráficos de dispersão confirmam visualmente esta conclusão: os pontos de dados formam uma nuvem difusa sem qualquer tendência ascendente ou descendente, e as linhas de regressão são praticamente horizontais.

Conclusão para RQ01: Os dados analisados não fornecem evidências para suportar a hipótese de que repositórios mais populares possuem melhor qualidade de código. A popularidade, por si só, não se mostrou um indicador confiável para as métricas de acoplamento, profundidade de herança ou coesão. Portanto, a hipótese **H1 é refutada**.

3.3 RQ02: Relação entre Maturidade e Qualidade

A segunda questão de pesquisa investiga se a maturidade de um projeto, definida pela sua idade em anos, tem relação com a qualidade do seu código. A hipótese (**H2**) sugere que repositórios mais maduros tendem a ter melhor qualidade, pois teriam passado por mais ciclos de refatoração e estabilização.

Para testar esta hipótese, os repositórios foram categorizados em três grupos de maturidade com base nos tercis de sua idade: "Jovens", "Intermediários"e "Maduros". Em seguida, foram gerados gráficos de box plot para comparar a distribuição das medianas de CBO, DIT e LCOM entre estes grupos, conforme ilustrado na Figura 5.

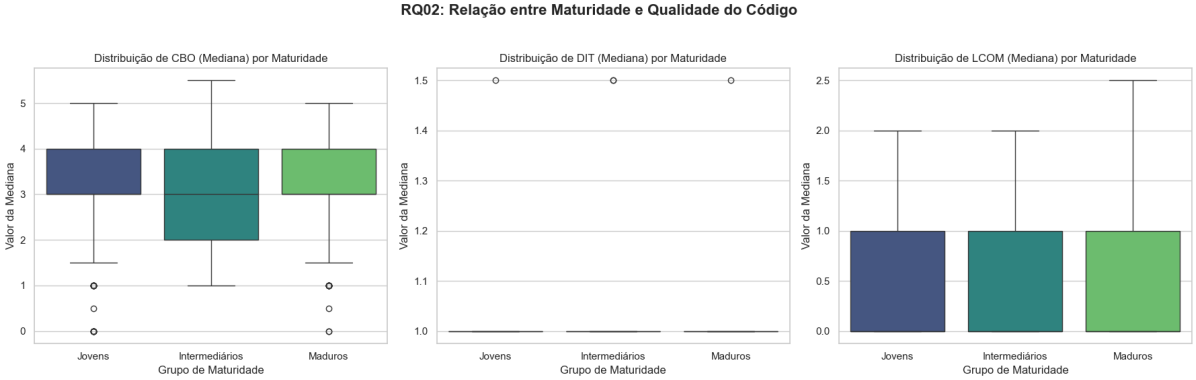

Figura 5: Box plots comparando a distribuição das métricas de qualidade entre grupos de repositórios com diferentes níveis de maturidade.

A análise visual dos box plots revela que não há uma tendência clara que suporte a hipótese formulada:

- Para o **CBO (Acoplamento)**, a mediana e a dispersão (tamanho da caixa) são muito semelhantes nos três grupos, indicando que a idade do projeto não tem um impacto aparente no nível de acoplamento das classes.
- Para o **DIT (Herança)**, observa-se uma ligeira tendência de aumento da mediana nos projetos mais maduros. No entanto, a sobreposição das caixas (intervalos interquartis) é muito grande, sugerindo que a diferença não é estatisticamente significativa.
- Para o **LCOM (Coesão)**, a mediana permanece praticamente constante nos três grupos, sem qualquer indicação de melhoria (diminuição do LCOM) em projetos mais antigos.

A semelhança notável na distribuição das métricas de qualidade entre os grupos de maturidade sugere que a idade, por si só, não é um fator determinante para a qualidade do código.

Conclusão para RQ02: Não foram encontradas evidências que sustentem a hipótese de que repositórios mais maduros apresentam melhor qualidade de código. As distribuições das métricas de CBO, DIT e LCOM são muito similares entre projetos jovens, intermediários e maduros. Portanto, a hipótese **H2 é refutada**.

3.4 RQ03: Relação entre Atividade e Qualidade

A terceira questão de pesquisa explora a relação entre a atividade de um repositório, medida pelo número de *releases*, e a qualidade do seu código. A hipótese (**H3**) postula que repositórios mais ativos tenderiam a ter uma qualidade inferior, devido à introdução constante de novas funcionalidades e correções rápidas (*hotfixes*) que poderiam levar à degradação da estrutura do código.

Para avaliar esta hipótese, os repositórios foram segmentados em três categorias de atividade ("Baixa", "Moderada"e "Alta") com base nos tercis do número total de *releases*. A Figura 6 apresenta os gráficos de violino que comparam a distribuição das métricas de qualidade para cada um destes grupos.

A análise dos gráficos revela uma tendência subtil que suporta a hipótese formulada:

- Para o **CBO (Acoplamento)**, observa-se que a mediana do grupo de "Alta"atividade é visivelmente superior à dos grupos de atividade "Baixa"e "Moderada". Isto indica que a alta frequência de alterações está correlacionada com um aumento no acoplamento entre as classes, um sinal de degradação da qualidade.
- Para o **DIT (Herança)** e **LCOM (Coesão)**, as medianas permanecem estáveis nos três grupos. Embora estas métricas não mostrem degradação, a tendência de aumento no acoplamento é um forte indicador de que a pressão por novas *releases* impacta negativamente a modularidade do código.

O aumento do acoplamento em repositórios mais ativos é um sinal claro de que a manutenção da qualidade se torna mais desafiadora sob um ciclo de desenvolvimento acelerado.

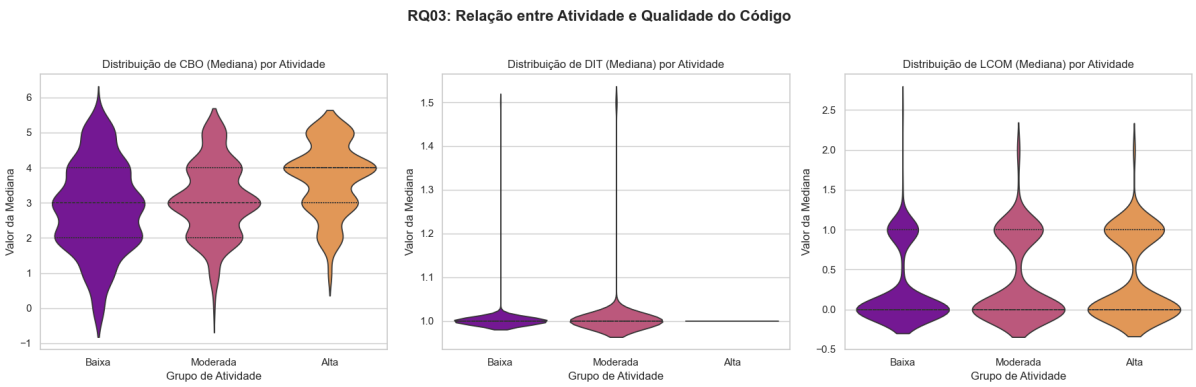

Figura 6: Gráficos de violino comparando a distribuição das métricas de qualidade entre grupos de repositórios com diferentes níveis de atividade.

Conclusão para RQ03: As evidências suportam a hipótese de que repositórios mais ativos possuem uma qualidade de código inferior. O aumento observado no acoplamento (CBO) no grupo de alta atividade é consistente com a ideia de que a pressão por entregas contínuas pode levar a um código menos modular e mais difícil de manter. Portanto, a hipótese **H3 é validada**.

3.5 RQ04: Relação entre Tamanho e Qualidade

A quarta e última questão de pesquisa aborda a relação entre o tamanho de um repositório, medido pelo somatório de Linhas de Código (LOC), e a sua qualidade. A hipótese (**H4**) propõe uma relação inversamente proporcional: quanto maior o repositório, menor tenderia a ser a sua qualidade, devido ao aumento da complexidade e do potencial acúmulo de débito técnico.

Para testar esta hipótese, os projetos foram divididos em três grupos de tamanho ("Pequenos", "Médios"e "Grandes") com base nos tercis da sua contagem total de LOC. A Figura 7 exibe os gráficos de barras que comparam a média das métricas de qualidade para cada um destes grupos.

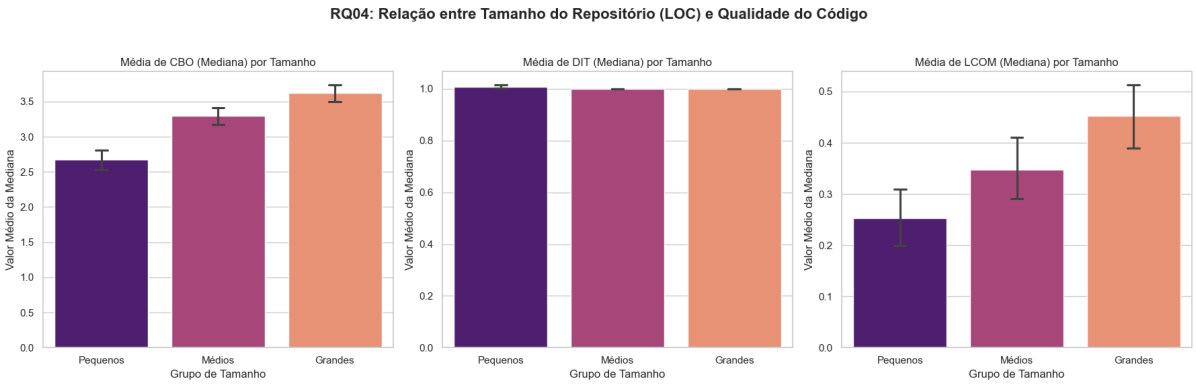

Figura 7: Gráficos de barras comparando a média das métricas de qualidade entre grupos de repositórios com diferentes tamanhos (LOC).

A análise dos gráficos revela tendências claras que sustentam a hipótese:

- Para o **CBO (Acoplamento)**, observa-se uma tendência consistente e progressiva de aumento na média à medida que o tamanho do projeto aumenta. Repositórios "Grandes"apresentam o maior nível de acoplamento, indicando uma qualidade estrutural inferior.
- Para o **DIT (Herança)**, a tendência é similar à do CBO, com um aumento claro na profundidade de herança média em projetos maiores. Isto sugere que a complexidade da hierarquia de classes cresce com o tamanho, o que pode dificultar a manutenção.
- Para o **LCOM (Coesão)**, embora a diferença na média não seja tão pronunciada, os valores não mostram uma melhoria da coesão em projetos maiores, o que, em conjunto com o aumento do CBO e DIT, reforça a ideia de que a qualidade não escala positivamente com o tamanho.

O aumento claro no acoplamento e na profundidade de herança é uma forte evidência de que a complexidade intrínseca de repositórios maiores se reflete negativamente em sua qualidade estrutural.

Conclusão para RQ04: A hipótese de que repositórios maiores possuem menor qualidade é suportada pelas tendências observadas. O aumento consistente do acoplamento (CBO) e da profundidade de herança (DIT) confirma que a qualidade do código tende a degradar-se à medida que a base de código cresce. Portanto, a hipótese **H4 é validada**.

4 Conclusão

Este estudo propôs-se a investigar a correlação entre métricas de processo de desenvolvimento (popularidade, maturidade, atividade e tamanho) e métricas de qualidade de código (acoplamento, herança e coesão) em 1.000 repositórios Java populares. As quatro hipóteses iniciais, baseadas em perceções comuns da comunidade de software, foram sistematicamente testadas.

Os resultados revelaram uma relação complexa entre o processo de desenvolvimento e a qualidade do código. Duas das quatro hipóteses foram refutadas, enquanto duas foram validadas:

- **H1 e H2 (Refutadas):** Não foram encontradas evidências de que a popularidade ou a maturidade de um repositório tenham uma relação direta com a qualidade do código.
- **H3 e H4 (Validadas):** Foram encontradas evidências que suportam as hipóteses de que repositórios mais ativos e maiores tendem a apresentar uma qualidade de código inferior, observada através de um aumento no acoplamento e, no caso do tamanho, na profundidade de herança.

4.1 Discussão dos Resultados

A ausência de correlação para popularidade e maturidade sugere que estes fatores, por si só, não garantem um código de alta qualidade. Por outro lado, a confirmação, mesmo que subtil, de que a atividade e o tamanho impactam negativamente a qualidade, reforça

a importância de práticas de engenharia de software robustas para gerir a complexidade. Projetos com muitas *releases* e uma grande base de código estão mais suscetíveis ao aumento do acoplamento, o que pode levar a um maior débito técnico se não for ativamente gerido.

É provável que a qualidade interna do código seja um fenómeno multifatorial, mas os resultados indicam que a pressão por novas entregas (atividade) e o crescimento do sistema (tamanho) são, de facto, fatores de risco para a degradação da qualidade estrutural do software.