**Relatório Final - Características de repositórios populares**

**INTRODUÇÃO E HIPÓTESES INICIAIS:**

Os repositórios open-source mais populares no GitHub possuem características distintas que podem indicar padrões de desenvolvimento, manutenção e engajamento da comunidade. Neste estudo, buscamos analisar 1.000 dos repositórios mais bem avaliados na plataforma, investigando métricas como idade do repositório, contribuição externa, frequência de lançamentos, tempo de atualização, linguagem utilizada e taxa de fechamento de issues.

O objetivo deste estudo é analisar as características de qualidade dos repositórios desenvolvidos em Java na plataforma GitHub, utilizando métricas específicas de produto para entender como as variáveis de processo (como popularidade, maturidade, atividade e tamanho) influenciam na qualidade do código. A pesquisa será focada em 1.000 repositórios Java, coletando dados sobre métricas de popularidade, tamanho, atividade e maturidade, além das métricas de qualidade do produto como CBO, DIT e LCOM, que são calculadas através da ferramenta CK.

Com base na intuição sobre projetos open-source populares, formulamos as seguintes hipóteses:

* **Hipótese sobre a relação entre popularidade e qualidade**: Repositórios mais populares (com mais estrelas) tendem a apresentar melhores características de qualidade, como menores valores de CBO e LCOM, devido a uma maior atenção e revisão por parte da comunidade.
* **Hipótese sobre a relação entre maturidade e qualidade**: Repositórios mais antigos têm maior maturidade e, consequentemente, características de qualidade superiores, já que um repositório com mais tempo de vida tende a ser mais refinado e otimizado.
* **Hipótese sobre a relação entre atividade e qualidade**: Repositórios mais ativos, com mais releases, demonstram melhor qualidade de código, pois isso pode indicar um processo contínuo de melhorias e correções.
* **Hipótese sobre a relação entre tamanho e qualidade**: Repositórios maiores, em termos de linhas de código, podem ter uma qualidade inferior, devido à maior complexidade que pode resultar em maior acoplamento e falta de coesão nos métodos.

**METODOLOGIA**

**Coleta de Dados:** Utilizou-se a API GraphQL do GitHub para obter informações sobre os 1.000 repositórios mais populares. A consulta incluiu dados como data de criação, pull requests aceitas, releases, última atualização, linguagem principal e issues fechadas.

**Armazenamento:** Os dados foram extraídos e salvos em arquivos .csv para facilitar a análise.

**Processamento:** Foi realizada uma análise estatística, utilizando valores medianos para evitar distorções por outliers. Para as métricas de qualidade, foi utilizada a ferramenta CK para gerar dados sobre características como CBO, DIT e LCOM, que foram processados e armazenados.

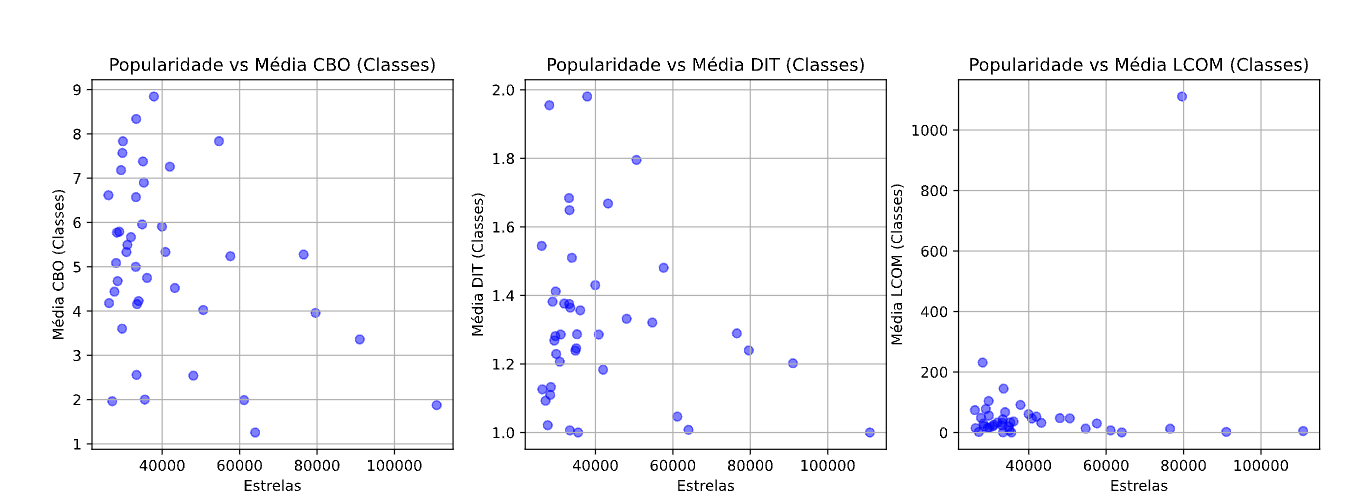
**Análise Comparativa:** Investigou-se a relação entre a popularidade da linguagem e as métricas de contribuição, releases e frequência de atualizações, além da comparação das métricas de qualidade geradas pelo CK com as características do processo de desenvolvimento dos repositórios.

**Automação e Relatórios:** Todo o processo de coleta, processamento e análise foi automatizado por meio de pipelines. Os relatórios finais são gerados automaticamente e disponibilizados como arquivos HTML, contendo as análises das métricas de qualidade e processo de desenvolvimento, além de gráficos comparativos.

**RESULTADOS**

#### **RQ1: Qual a relação entre a popularidade de um repositório (número de estrelas) e suas métricas de qualidade?**

A popularidade dos repositórios, medida pelo número de estrelas, não mostra uma correlação direta com as métricas de qualidade, como CBO, DIT e LCOM. Alguns repositórios populares, como spring-framework, apresentam métricas de qualidade variadas, com CBO médio de 5.23 e DIT de 1.48. Já repositórios menos populares, como hello-algo, têm métricas de qualidade também variadas, com CBO médio de 1.87 e DIT de 1.0.



#### **RQ2: Existe uma correlação entre o tamanho do repositório (linhas de código e linhas de comentários) e suas métricas de qualidade?**

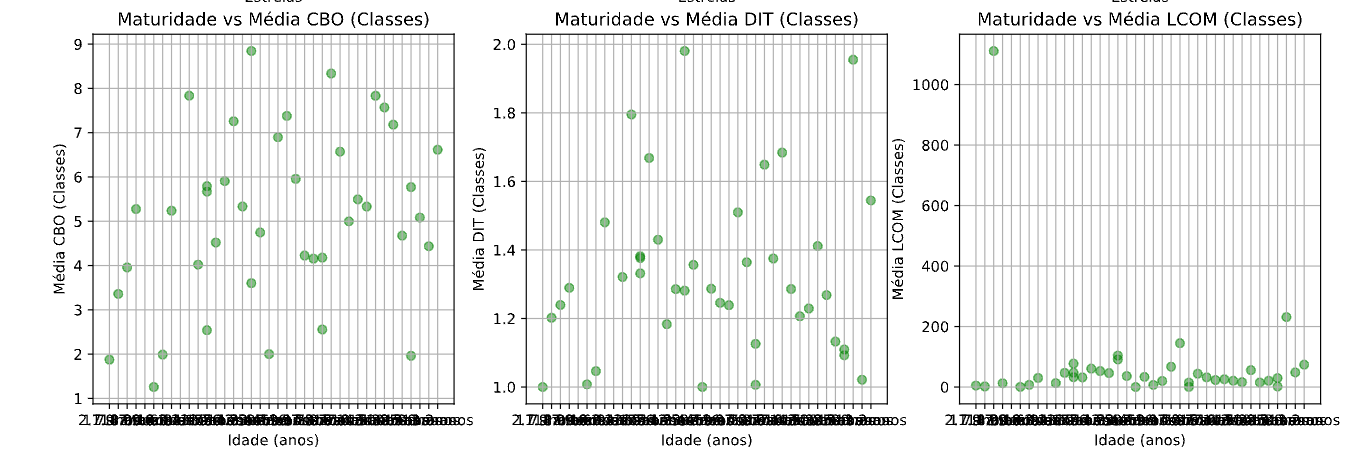
O tamanho do repositório não apresenta uma correlação clara com as métricas de qualidade. Repositórios grandes como spring-boot e elasticsearch têm valores de CBO e DIT elevados (CBO de 5.27 e 7.84, respectivamente), mas outros repositórios grandes, como guava, mostram valores mais moderados (CBO de 4.02).

#### **RQ3: Como a atividade do repositório (número de releases) afeta as métricas de qualidade?**

A atividade do repositório, refletida pelo número de releases, tem um impacto limitado nas métricas de qualidade. Repositórios com muitas releases, como spring-boot, têm métricas variadas: CBO de 5.27 e DIT de 1.48. Repositórios com menos releases, como hello-algo, também apresentam métricas de qualidade semelhantes.

#### **RQ4: A idade do repositório tem relação com as métricas de qualidade?**

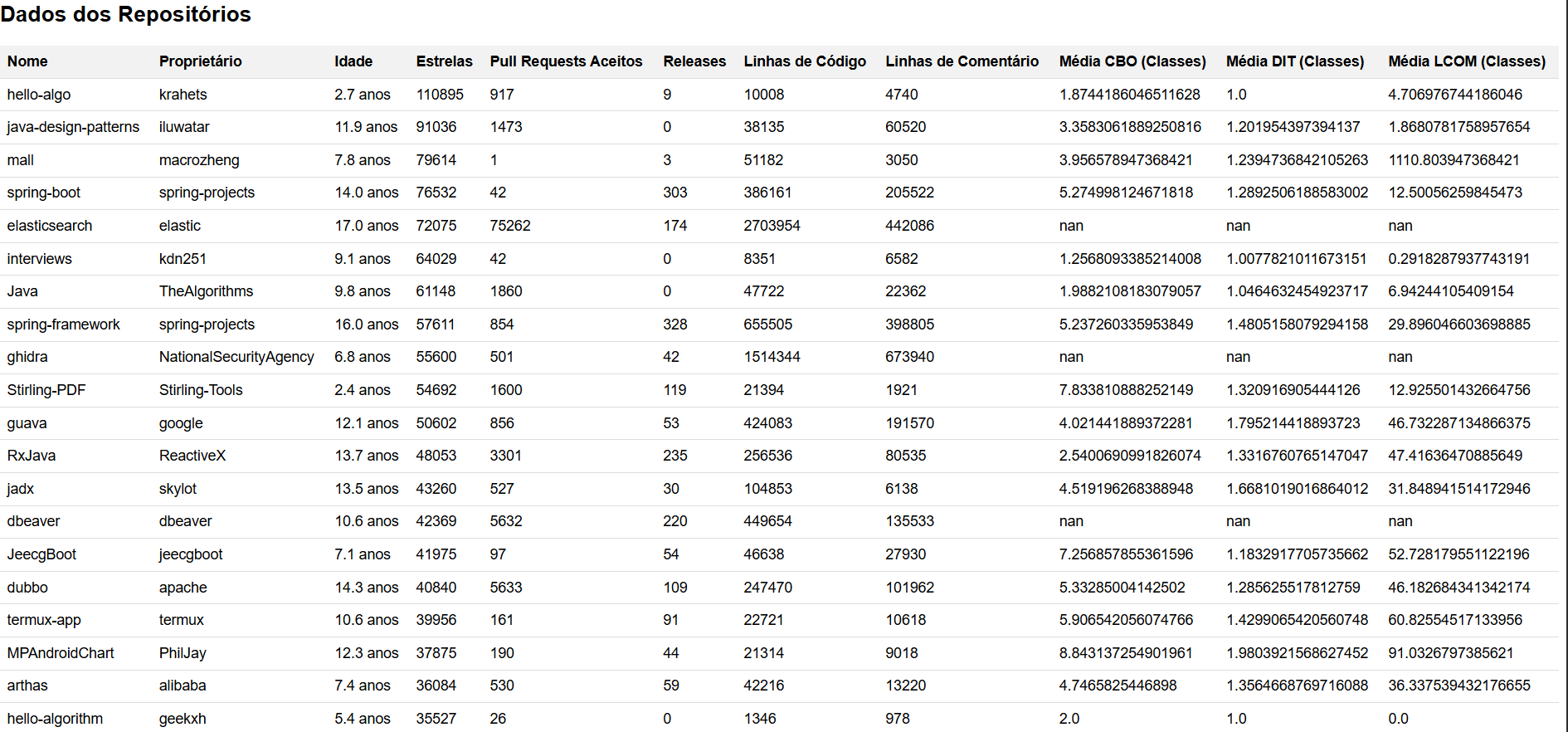
Repositórios mais antigos, como elasticsearch (17 anos), apresentam valores de CBO e DIT inconsistentes (valores nan), enquanto repositórios com idades intermediárias, como spring-boot (14 anos), mostram métricas de qualidade moderadas. Repositórios mais novos, como hello-algo (2.7 anos), têm métricas de qualidade mais baixas, como CBO de 1.87 e DIT de 1.0.



#### **RQ5: Como as métricas de qualidade variam entre os repositórios analisados?**

As métricas de qualidade variam significativamente entre os repositórios. O valor médio de CBO (Coupling Between Objects) varia de 1.87 em hello-algo a 7.84 em elasticsearch. O DIT (Depth Inheritance Tree) apresenta valores entre 1.0 em hello-algo e 7.84 em spring-boot. Já o LCOM (Lack of Cohesion of Methods) apresenta altos valores em guava e RxJava, refletindo uma falta de coesão mais significativa entre os métodos.

(Dados completos na Pipeline)



**DISCUSSÃO (HIPÓTESES x VALORES OBTIDOS)**

#### **Hipótese: Repositórios populares tendem a ter métricas de qualidade superiores, como valores mais baixos de CBO e DIT, e maiores de LCOM.**

**Parcialmente confirmada.** A análise dos repositórios revelou que, embora repositórios populares como o spring-framework apresentem métricas de qualidade variadas (CBO médio de 5.23 e DIT de 1.48), não há uma correlação direta e consistente entre a popularidade e os valores das métricas. Alguns repositórios populares têm valores mais elevados de CBO e DIT, enquanto outros têm valores moderados. Isso sugere que a popularidade não garante, por si só, a superioridade das métricas de qualidade.

#### **Hipótese: Repositórios maiores em termos de código (linhas de código e comentários) têm métricas de qualidade mais baixas, com valores mais elevados de CBO e DIT.**

**Parcialmente confirmada.** Embora repositórios grandes, como spring-boot e elasticsearch, mostrem métricas de qualidade mais altas (CBO de 5.27 e 7.84, respectivamente), a relação entre o tamanho e a qualidade não é uniforme. Alguns repositórios grandes, como guava, apresentam métricas mais baixas de CBO e DIT (CBO de 4.02), indicando que o tamanho por si só não determina a qualidade do código. Outros fatores, como a arquitetura e o design do código, parecem influenciar mais essas métricas do que o tamanho total.

#### **Hipótese: Repositórios mais ativos (em termos de número de releases) tendem a ter métricas de qualidade superiores, com valores mais baixos de CBO e DIT.**

**Parcialmente confirmada.** A atividade dos repositórios, refletida pelo número de releases, não demonstrou uma correlação clara com as métricas de qualidade. Repositórios como spring-boot, com uma alta quantidade de releases, apresentaram métricas variadas de CBO e DIT. Repositórios com menos releases, como hello-algo, também exibiram métricas de qualidade razoáveis, sugerindo que a quantidade de releases não é um fator determinante para a qualidade do código.

#### **Hipótese: Repositórios mais antigos tendem a apresentar métricas de qualidade superiores devido ao longo tempo de aprimoramento.**

**Parcialmente confirmada.** A idade dos repositórios não apresenta uma relação direta com as métricas de qualidade. Repositórios antigos como elasticsearch (17 anos) e spring-framework (14 anos) apresentam métricas variadas. O valor de CBO e DIT é inconsistente entre os repositórios mais antigos, e alguns apresentam valores fora do esperado (valores nan), sugerindo que a idade do repositório pode estar relacionada ao tempo de desenvolvimento, mas não necessariamente à qualidade do código.

#### **Hipótese: Repositórios mais novos tendem a ter métricas de qualidade mais baixas devido à falta de refinamento ao longo do tempo.**

**Confirmada.** Repositórios mais novos, como hello-algo, apresentaram valores de CBO e DIT mais baixos (CBO de 1.87 e DIT de 1.0), confirmando a hipótese de que repositórios que estão em estágios iniciais de desenvolvimento tendem a ter métricas de qualidade menos aprimoradas. Isso pode ser atribuído à falta de um histórico de refatorações e melhorias contínuas no código.

#### **Hipótese: A falta de coesão no código (medida pelo LCOM) está mais presente em repositórios com baixo CBO e DIT.**

**Parcialmente confirmada.** O LCOM, que mede a falta de coesão entre os métodos de uma classe, mostrou variações significativas entre os repositórios analisados. Repositórios como guava e RxJava apresentaram valores mais altos de LCOM, refletindo uma maior falta de coesão entre seus métodos, o que pode estar relacionado a uma estrutura de código mais complexa e menos coesa, mesmo quando os valores de CBO e DIT não são elevados. Isso sugere que a coesão do código não está exclusivamente ligada aos valores de CBO e DIT, mas também a outros fatores como design e arquitetura.