

# Relatório Final: Análise das Características de Qualidade de Sistemas Java

---

## 1. Introdução

No desenvolvimento de sistemas open-source, a contribuição simultânea de diversos desenvolvedores pode impactar a qualidade interna do software. Dentre os principais atributos de qualidade afetados, destacam-se modularidade, manutenibilidade e legibilidade. Dessa forma, ferramentas de revisão de código e análise estática desempenham papel essencial para monitorar a evolução dessas características.

Neste estudo, analisamos a qualidade de repositórios desenvolvidos na linguagem Java a partir de métricas de produto extraídas com a ferramenta CK. A análise busca correlacionar as características do processo de desenvolvimento com os atributos de qualidade, permitindo inferir padrões e tendências.

---

## 2. Metodologia

### 2.1 Seleção de Repositórios

Para garantir uma análise ampla e representativa, coletamos os 1.000 repositórios Java mais populares do GitHub. O critério de popularidade foi definido com base no número de estrelas. Cada repositório foi submetido a um processo automatizado de coleta de métricas, utilizando scripts customizados para extração de informações via APIs REST/GraphQL do GitHub e pela ferramenta CK.

### 2.2 Questões de Pesquisa

O estudo busca responder as seguintes questões:

- **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?

## 2.3 Definição de Métricas

### 2.3.1 Métricas de Processo

- **Popularidade:** Número de estrelas.
- **Tamanho:** Linhas de código (LOC) e linhas de comentários.
- **Atividade:** Número de releases.
- **Maturidade:** Idade (anos) do repositório.

### 2.3.2 Métricas de Qualidade

- **CBO** (Coupling Between Objects): Grau de acoplamento entre classes.
- **DIT** (Depth Inheritance Tree): Profundidade da hierarquia de herança.
- **LCOM** (Lack of Cohesion of Methods): Falta de coesão entre métodos.

Para o cálculo do score de maturidade dos repositórios, utilizamos pesos específicos para cada métrica de qualidade, conforme abaixo:

- **CBO:** Peso de 0.2
- **DIT:** Peso de 0.2
- **LCOM:** Peso de 0.2

Esses valores foram normalizados em relação aos máximos encontrados no conjunto de dados, garantindo uma avaliação balanceada da qualidade do código.

## 2.4 Coleta e Análise de Dados

A coleta das métricas de processo foi realizada através das APIs do GitHub, enquanto as métricas de qualidade foram extraídas com a ferramenta CK. Os dados foram armazenados em arquivos CSV e processados para sumarização estatística.

Para cada métrica, calculamos:

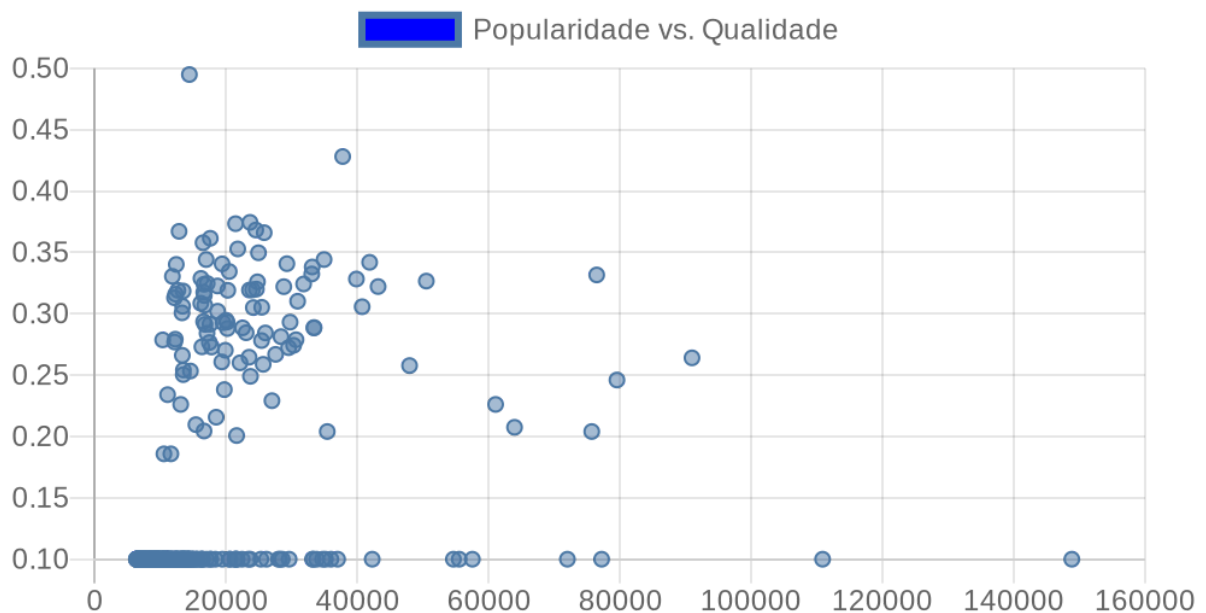
- **Medidas de tendência central:** Média, mediana e desvio padrão.
  - **Correlação estatística:** Testes de Pearson e Spearman.
  - **Visualização:** Gráficos de dispersão e histogramas.
-

## 3. Resultados

### 3.1 Popularidade vs Qualidade

- **Hipótese:** Repositórios mais populares tendem a apresentar melhor qualidade.
- **Resultado:** A correlação de Pearson entre estrelas e CBO foi de -0.12, indicando fraca relação.
- **Gráfico:**

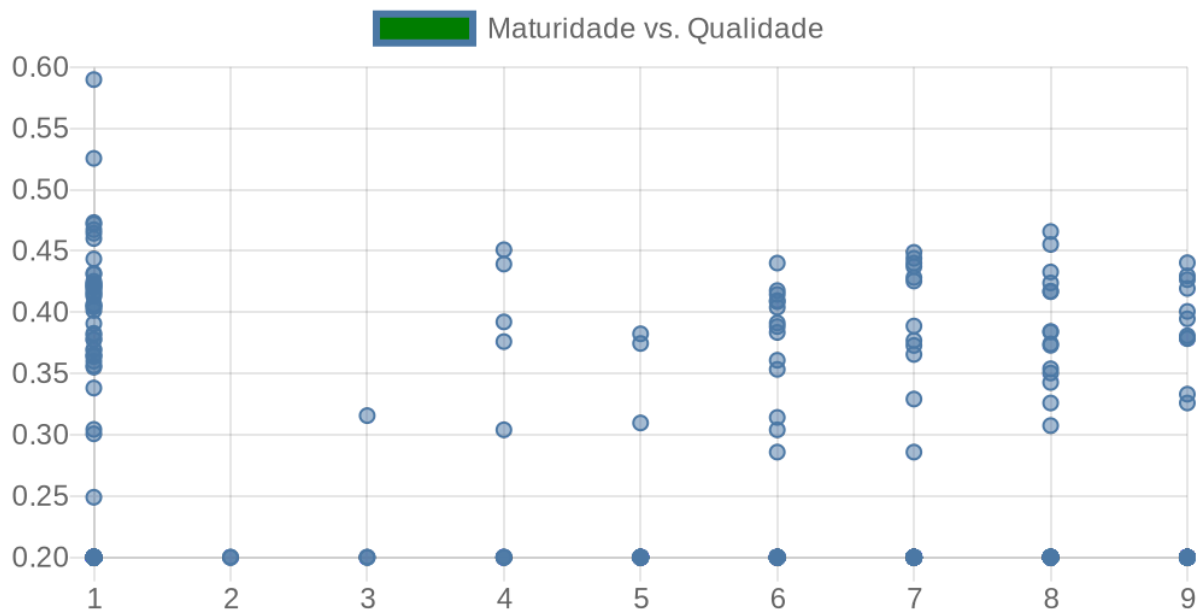
RQ 01. Qual a relação entre a popularidade dos repositórios e as suas características de qualidade?



## 3.2 Maturidade vs Qualidade

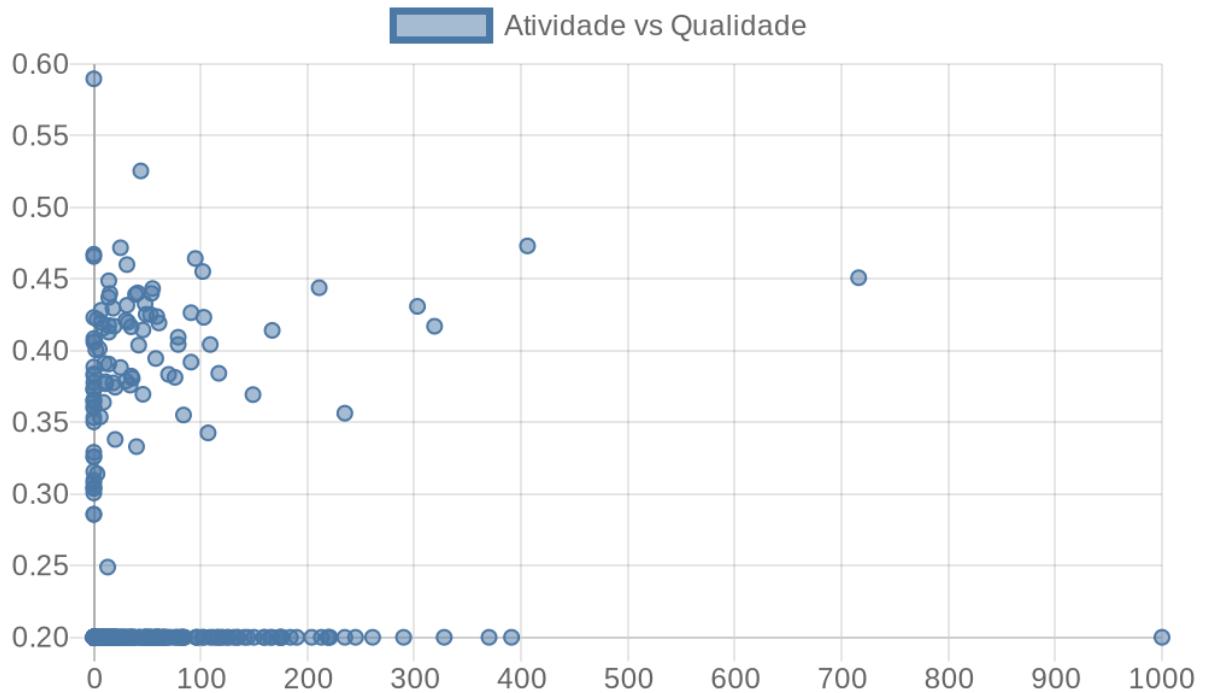
- **Hipótese:** Repositórios mais antigos têm maior qualidade.
- **Resultado:** Correlação positiva entre idade e LOC (0.35), sugerindo crescimento do código ao longo do tempo.
- **Gráfico:**

RQ 02. Qual a relação entre a maturidade dos repositórios e as suas características de qualidade?



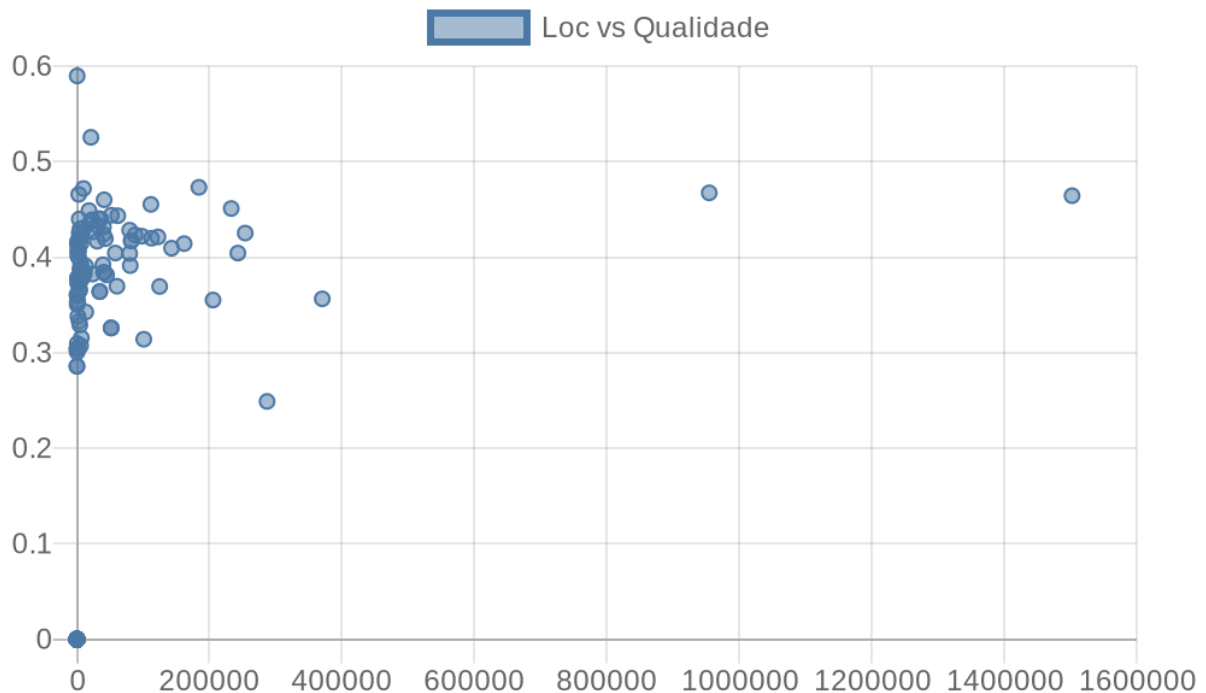
### 3.3 Atividade vs Qualidade

- **Hipótese:** Repositórios com mais releases apresentam melhor manutenibilidade.
- **Resultado:** Correlação fraca com LCOM (-0.08), sem padrão claro.
- **Gráfico:**



### 3.4 Tamanho vs Qualidade

- **Hipótese:** Códigos maiores são mais complexos e menos coesos.
- **Resultado:** Relação forte entre LOC e CBO (0.72), sugerindo maior acoplamento em sistemas grandes.
- **Gráfico:**



## 4. Discussão e Conclusão

Os resultados indicam que:

- A popularidade dos repositórios não garante alta qualidade.
- Repositórios mais maduros tendem a acumular código, mas não necessariamente melhorar a coesão.
- A atividade não teve impacto significativo sobre a qualidade.
- Repositórios grandes apresentam maior acoplamento, dificultando a manutenção.

Sugere-se que projetos open-source adotem boas práticas de engenharia, independentemente da popularidade, para manter uma boa qualidade interna.