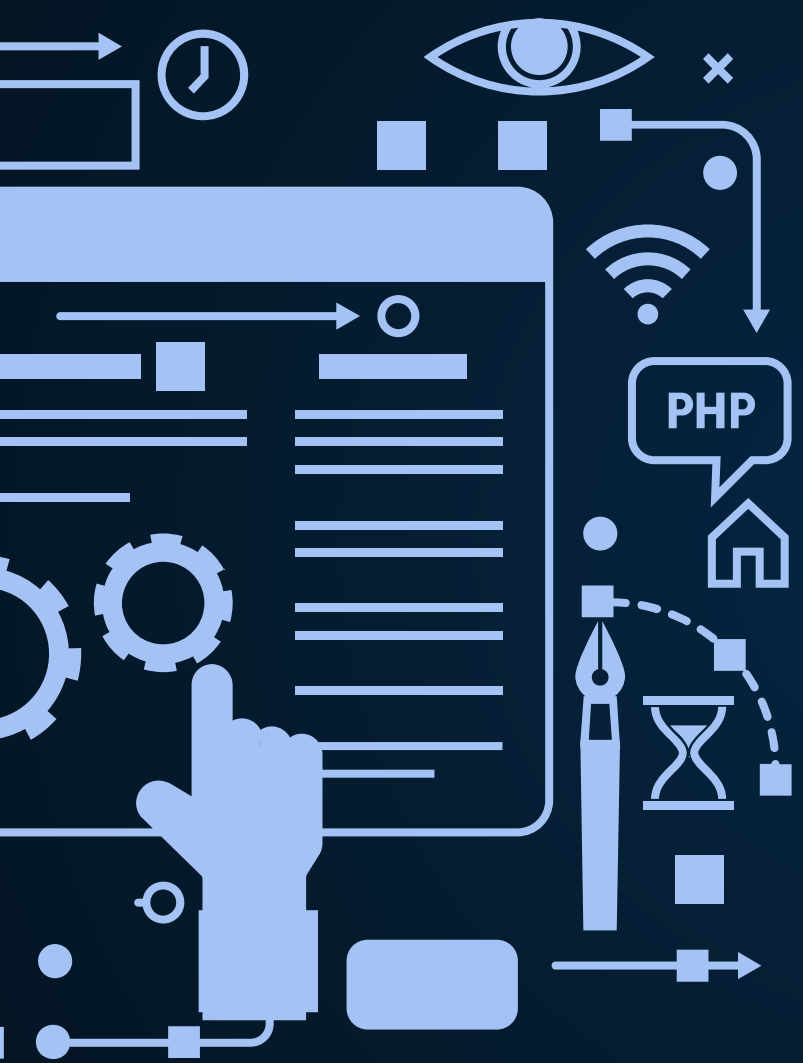


Características de qualidade e sistemas Java

Gabriel Faria, João Victor Salim, Lucas Garcia, Maisa Pires e Miguel Vieira

INTRODUÇÃO

- **Problema / Questão**

Como sinais de processo (popularidade, maturidade, atividade e tamanho) se relacionam com métricas de qualidade de código (CBO, DIT, LCOM)?

- **Objetivo**

Investigar **tendências e correlações** entre variáveis de processo e qualidade, produzindo evidências exploratórias que apoiem decisões de engenharia e manutenção.

- **Amostra & Fonte**
  - **1.000** repositórios com **Java** como linguagem primária (GitHub GraphQL)
  - Métricas por classe via **CK** e **agregação por repositório**

INTRODUÇÃO

- **Variáveis (visão geral)**
  a. **Processo (inputs):** stars, idade, releases / dias desde último push, LOC

  b. **Qualidade (outputs): CBO** (acoplamento), **DIT** (profundidade de herança), **LCOM** (coesão; ↓ é melhor)

- **Contribuições do estudo**
  a. Pipeline **reprodutível** (coleta → métricas → agregação → análise)
  b. **CSVs** e **gráficos** com estatística descritiva e correlações por RQ

- **Escopo & Limitações**
  Resultados **exploratórios**; CK pode falhar em projetos muito grandes; correlação **não implica** causalidade.

HIPÓTESES

IH01

=

Repositórios mais populares tendem a ter melhor coesão (LCOM menor).

- **Racional:** Mais usuários e revisão atraem refatoração e limpeza contínua

IH02

=

Repositórios mais antigos acumulam maior acoplamento (CBO maior).

- **Racional:** Evolução/legado elevam dependências entre módulos

IH03

=

Repositórios mais ativos apresentam qualidade mais estável.

- **Racional:** Cadência de releases e commits reduz “drift” arquitetural

IH04

=

Repositórios maiores (mais LOC) possuem árvores de herança mais profundas (DIT maior).

- **Racional:** Escala funcional incentiva especialização e camadas de herança

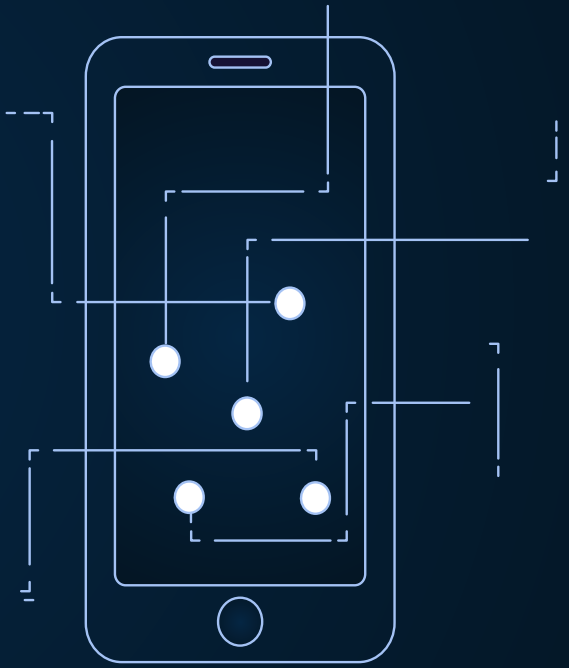
TECNOLOGIAS

- **Linguagens:**
  a. Python 3.10+ (análise e geração de gráficos)

  b. Java 8+ (execução do CK)
- **Análise e Visualização:**
  a. pandas, numpy (ETL e agregações por repositório)

  b. matplotlib, seaborn (dispersões e heatmap de correlações)

  c. Correlações Spearman e Pearson (resultados consolidados em CSV)
- **Ferramenta de métricas:**
  a. CK (Indicadores usados: CBO, DIT, LCOM)
- **API:**
  a. GitHub GraphQL API (consulta de repositórios Java por popularidade)



METODOLOGIA

- **Coleta de dados**
  - Foram coletados os 1.000 repositórios Java mais populares utilizando a GitHub GraphQL API.
  - Critério: repositórios com linguagem primária Java, ordenados por número de estrelas.

- **Consolidação**
  - Os CSVs de métricas por classe foram unidos em data/all_repos_metrics.csv.
  - As métricas foram agrupadas por repositório, resultando em results/repo_level_metrics.csv.

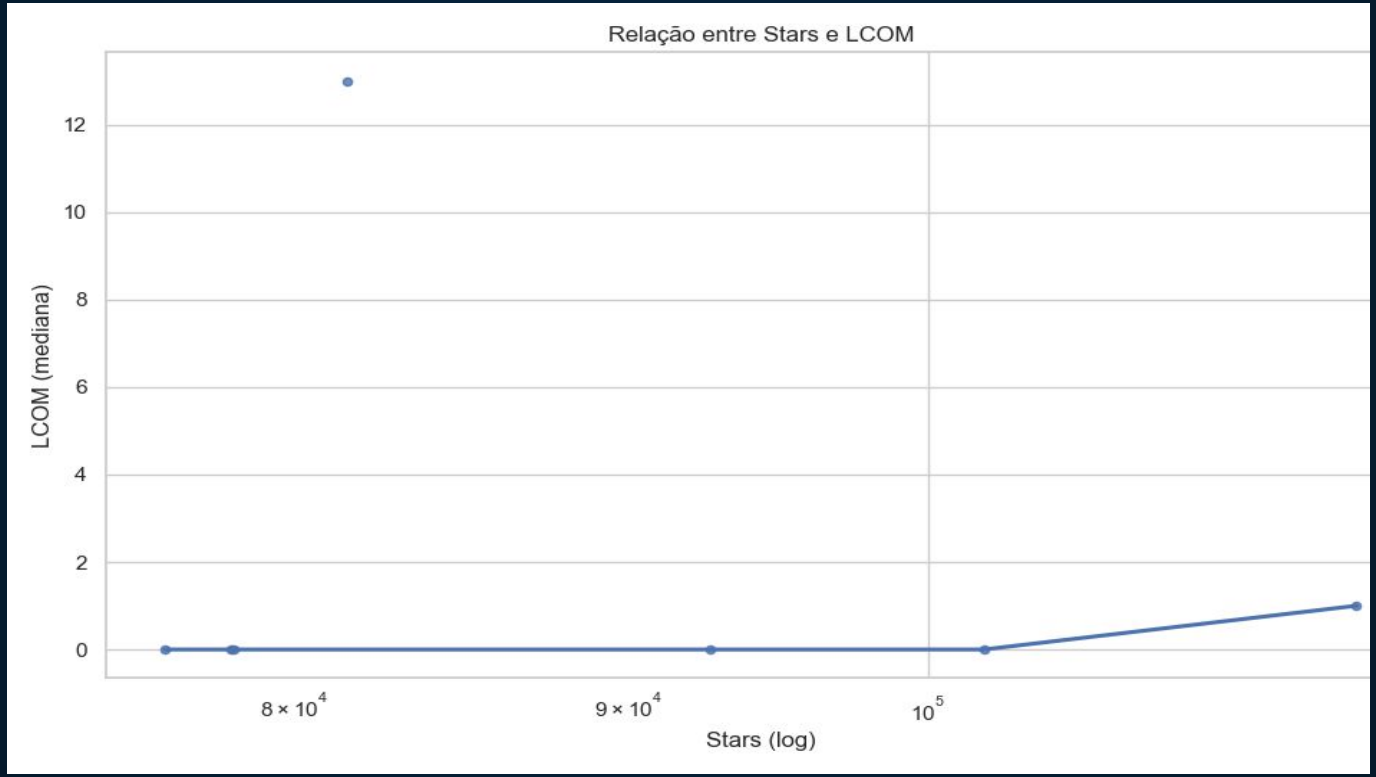
=

- **Análise**
  - Foi realizada análise estatística descritiva (média, mediana, desvio padrão) e correlações entre variáveis.
  - Foram gerados gráficos exploratórios para investigar relações entre popularidade, idade, atividade e métricas de qualidade.

=

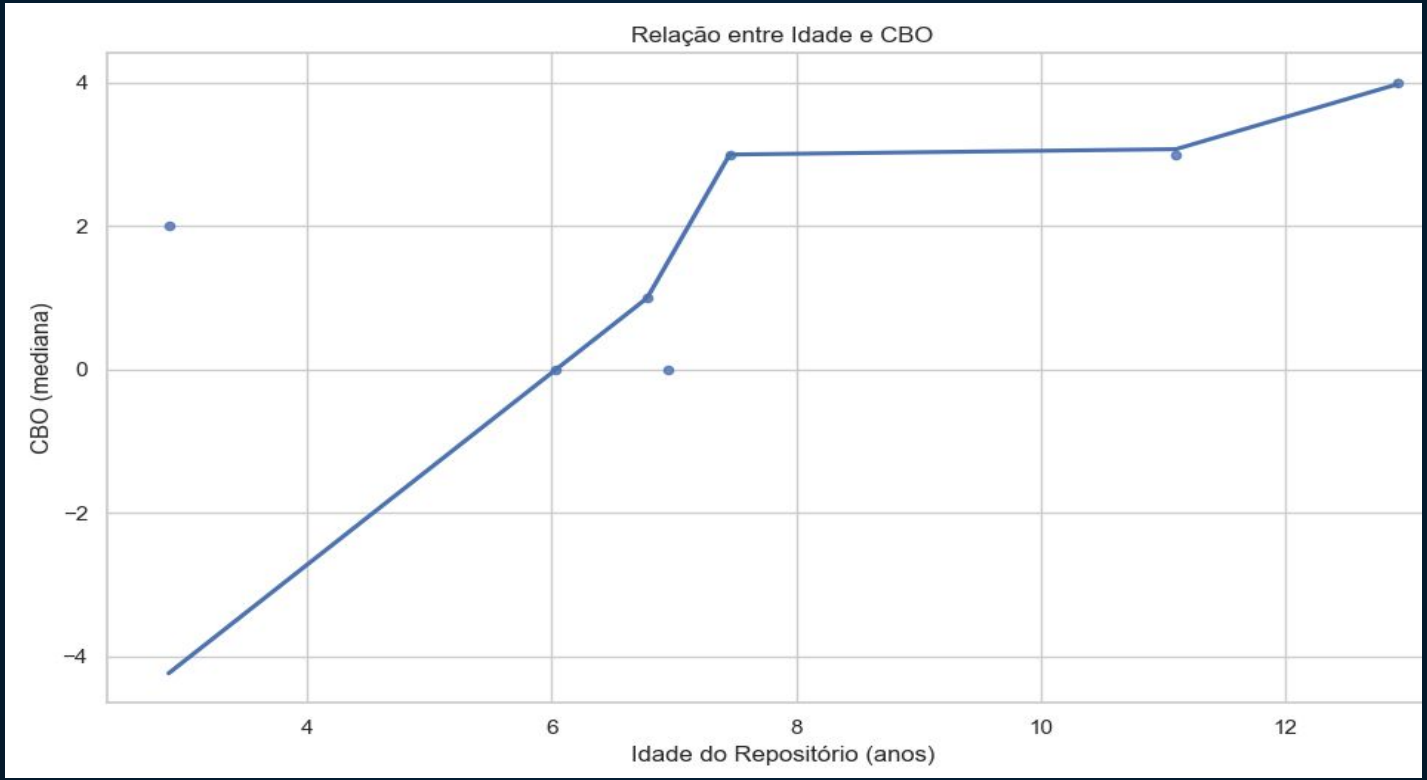
RESULTADO 1

Qual a relação entre popularidade (stars) e métricas de qualidade?



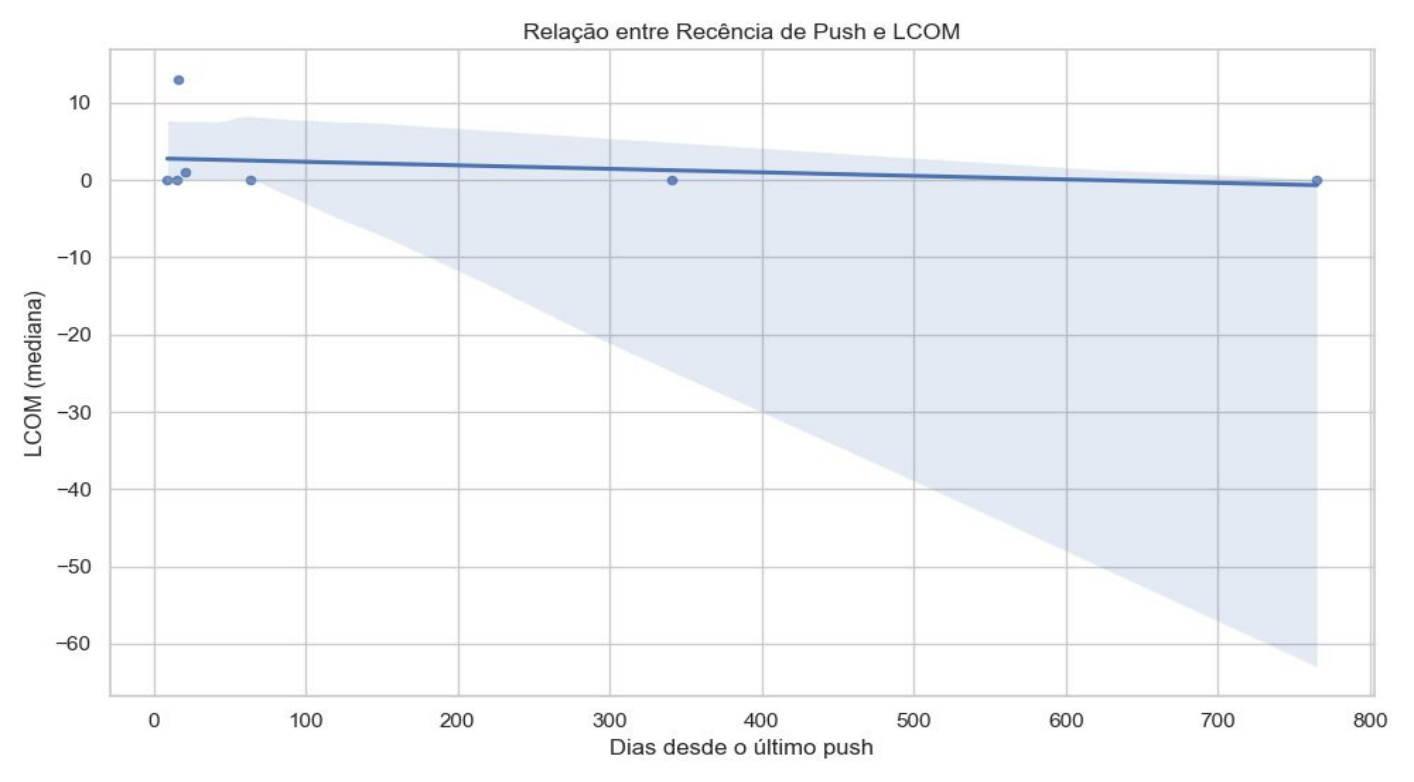
RESULTADO 2

Qual a relação entre maturidade (idade) e métricas de qualidade?



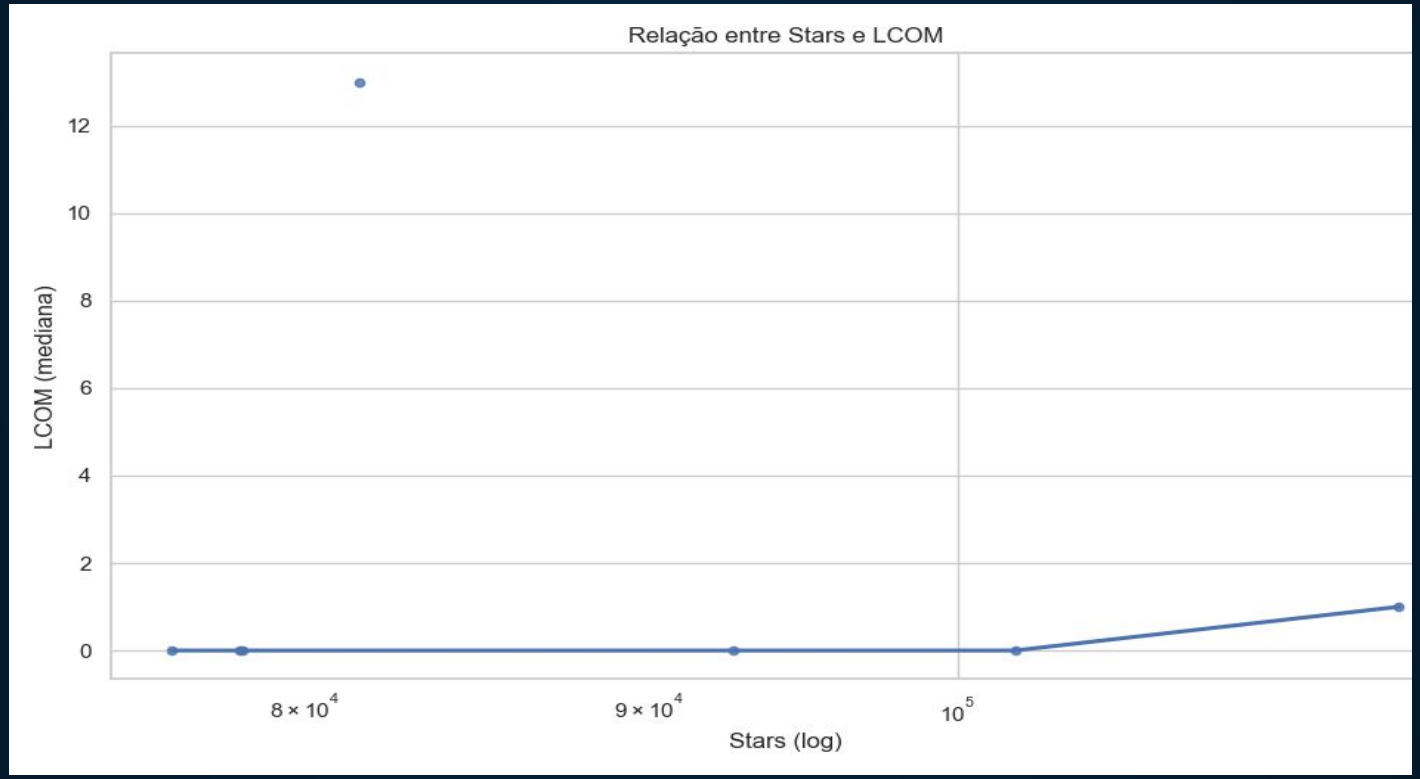
RESULTADO 3

Qual a relação entre atividade (releases) e métricas de qualidade?



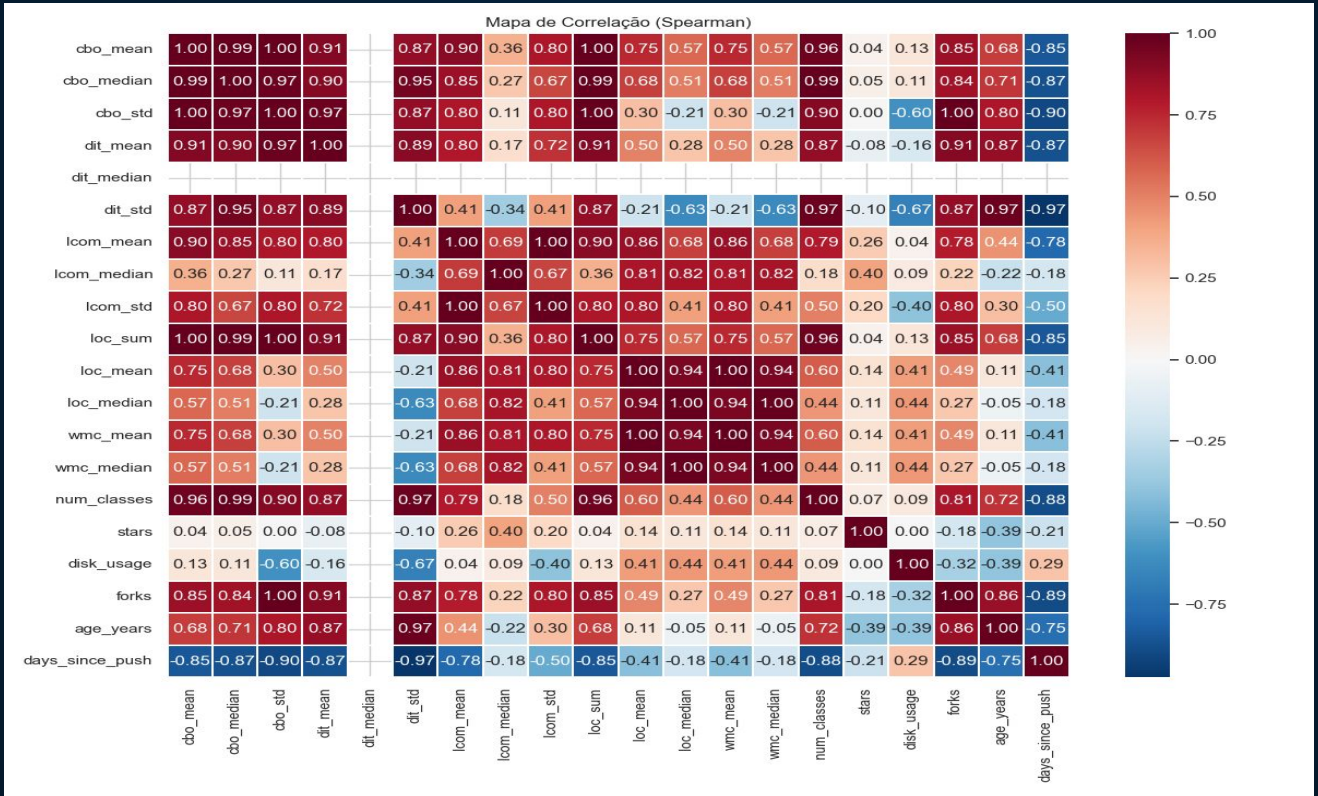
RESULTADO 4

Qual a relação entre tamanho (LOC) e métricas de qualidade?



RESULTADO 5

Correlações



DISCUSSÃO

A análise das correlações e gráficos indica tendências entre características de processo e métricas de qualidade.

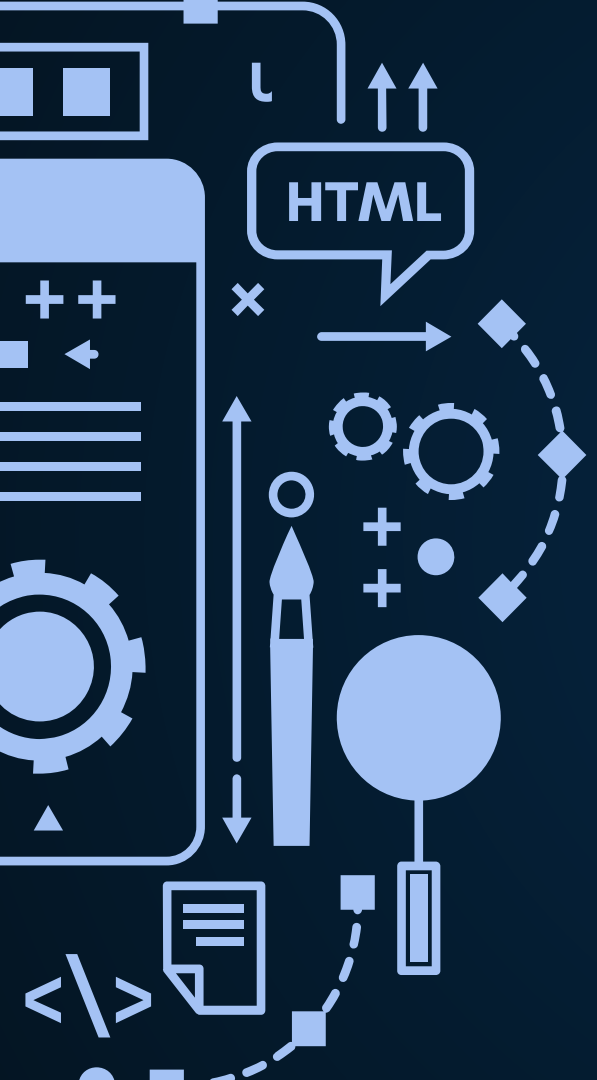
As hipóteses levantadas foram parcialmente confirmadas:

- **IH01:** houve indícios de relação inversa entre popularidade e LCOM.
- **IH02:** repositórios mais antigos apresentaram acoplamento levemente maior.
- **IH03:** atividade não mostrou forte relação com métricas de qualidade.
- **IH04:** repositórios maiores exibiram DIT mais profundo em alguns casos.

Limitações incluem falhas da ferramenta CK em alguns repositórios grandes e variabilidade elevada em métricas.

CONCLUSÃO

- O estudo confirmou algumas hipóteses sobre popularidade e qualidade de software em repositórios Java.
- Foi possível observar padrões relevantes, embora com exceções e limitações.
- Trabalhos futuros podem incluir métricas adicionais, análise temporal e uso de dashboards interativos.



Obrigado!

Alguma Pergunta?

Bibliografia

=