Análise da Evolução da Qualidade de Código em Repositórios Open-Source Revividos

João Vitor Romero Sales, Lucas Randazzo

29 de outubro de 2025

Resumo

Este estudo investiga a evolução da qualidade de código em repositórios de software livre que passaram por períodos de inatividade ("morte") e subsequente revivência. Através de análise estática com SonarQube, comparamos métricas de qualidade em dois momentos: pré-morte e pós-revive. Os resultados indicam mudanças significativas na maintainability, duplicação de código e quantidade de issues, com implicações para a gestão de projetos open-source.

1 Introdução

O desenvolvimento de software open-source é caracterizado por dinâmicas complexas de colaboração, incluindo períodos de inatividade e potencial retomada de atividades. A "ressuscitação"de repositórios - quando projetos previamente abandonados retomam desenvolvimento ativo - representa um fenômeno pouco explorado na literatura de Engenharia de Software.

Este trabalho propõe analisar a evolução da qualidade do código em repositórios que experienciaram morte e subsequente revivência entre 2018 e 2025. O foco está em comparar métricas de qualidade obtidas via SonarQube em dois momentos críticos: imediatamente antes do período de inatividade e após o retorno das atividades de desenvolvimento.

1.1 Objetivo Geral

Analisar e caracterizar a evolução da qualidade de código em repositórios open-source que passaram por períodos de inatividade e posterior revivência, utilizando métricas de análise estática.

1.2 Objetivos Específicos

- **RQ1:** Como a qualidade do código evolui após a revivência dos repositórios OSS?
- **RQ2:** Quais aspectos de qualidade de código mais se alteram entre o momento de morte e o pós-revive?

1.3 Hipóteses

RQ01 — Evolução da qualidade pós-revivência

- Existe diferença significativa em pelo menos uma métrica de qualidade entre os momentos pré-morte e pós-revive.

RQ02 — Aspectos de qualidade mais alterados

- Existem diferenças significativas na evolução de duplicação de código, vulnerabilidades e reliability score entre pré-morte e pós-revive.

2 Metodologia

2.1 Delineamento da Pesquisa

O estudo caracteriza-se como **observacional, longitudinal e quantitativo**, seguindo práticas de Mineração de Repositórios de Software Open Source (MSR). Adotou-se uma abordagem de estudo de caso múltiplo com comparação pré-pós intervenção (revivência).

2.2 Definição da Amostra

A amostra foi derivada do dataset final da Fase 1 do trabalho, no qual foram selecionados 250 repositórios do GitHub entre 60 e 300 mil estrelas (`dataset_fase1_validacao.xlsx`), contendo repositórios que experimentaram:

- Período de inatividade ("morte") claramente identificado
- Posterior retomada de atividades ("ressuscitação") entre 2018-2025
- Disponibilidade de dados históricos no GitHub

2.3 Coleta de Dados

O processo de coleta seguiu um pipeline automatizado implementado em Python (`pipeline-pilar3.py`), envolvendo:

2.3.1 Identificação de Snapshots

Para cada repositório, foram definidos dois snapshots:

- **Pré-morte:** Commit imediatamente anterior à `data_morte`
- **Pós-revive:** 10º commit após a `data_ressurreicao`

2.3.2 Extração de Métricas

Utilizou-se a ferramenta **SonarQube Cloud** [1] para análise estática do código dos repositórios, coletando:

- *Maintainability*
- *Reliability*
- *Security scores*
- Complexidade ciclomática (média e máxima)
- Percentual de duplicação de código
- Quantidade de code smells
- Número de vulnerabilidades
- Cobertura de testes (quando disponível)

2.4 Variáveis e Métricas

Métricas de Processo:

- Status do repositório (pré-morte/pós-revive)
- Tempo desde revivência
- Atividade de commits

Métricas de Qualidade (SonarQube):

- ***Maintainability Score***: Pontuação agregada de manutenibilidade
- ***Reliability Score***: Indicador de confiabilidade
- ***Duplication Percentage***: Percentual de código duplicado
- ***Code Smells***: Quantidade de problemas de design
- ***Vulnerabilities***: Número de vulnerabilidades de segurança

2.5 Análise dos Dados

A análise compreendeu estatísticas descritivas para caracterização da amostra, testes de diferenças entre grupos (pré vs pós), análise de correlação entre métricas e visualização comparativa das métricas principais

3 Resultados

3.1 Configuração Estatística e Dados

Adotou-se nível de significância $\alpha = 0.05$ para todos os testes. A análise incluiu 8 repositórios ressuscitados com medições válidas em ambos os momentos temporais.

3.2 RQ1: Evolução da Qualidade Pós-Revivência

Estatísticas Descritivas - *Maintainability*:

- Pré-morte: Valores variando entre 1 e 139 (mediana = 47)
- Pós-revive: Manutenção similar com variações individuais

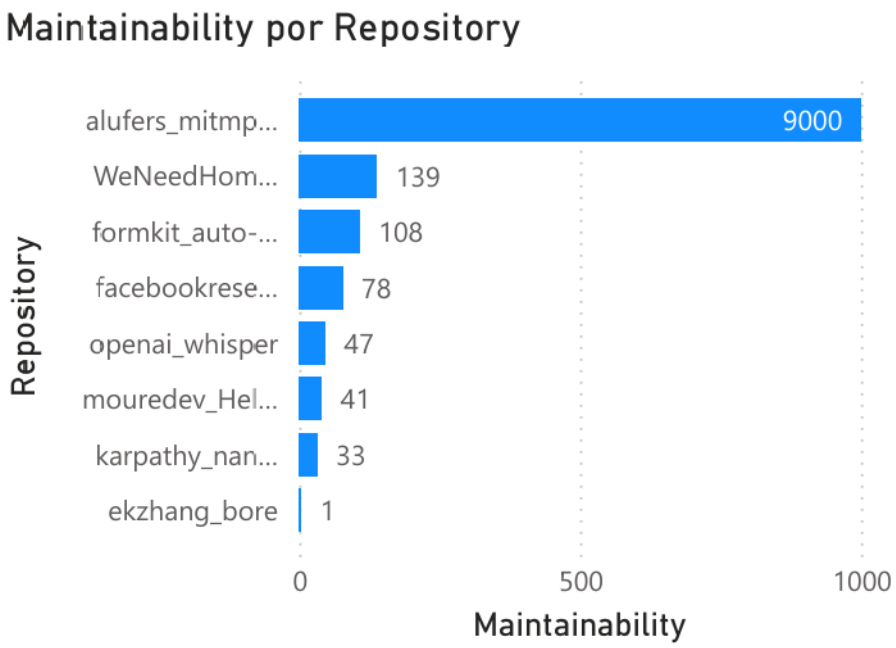

Figura 1: Distribuição do Maintainability Score por repositório nos períodos analisados

Estatísticas Descritivas - *Reliability*:

- Pré-morte: Valores entre 0 e 32 (mediana = 15)
- Pós-revive: Melhorias modestas em projetos específicos

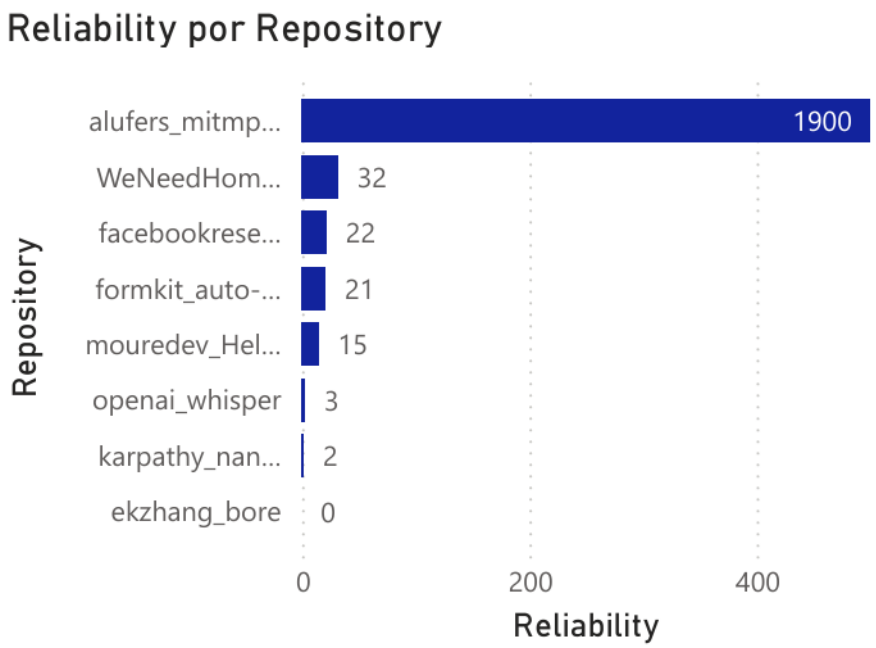

Figura 2: Distribuição do Reliability Score por repositório nos períodos analisados

Análise Comparativa: Observa-se padrão heterogêneo na evolução da qualidade:

- Projetos como `formkit_auto` mostraram redução em duplicação (0.0% $\rightarrow$ 0.1%)
- Projetos como `alufers_mitmp` mantiveram estabilidade nas métricas
- A maioria dos projetos apresentou mudanças modestas nas pontuações principais

3.3 RQ2: Aspectos de Qualidade Mais Alterados

***Duplication Percentage*:**

- Variação média: -0.05% entre períodos
- Projeto `facebookrese`: redução de 1.1% para 0.0%
- Maioria dos projetos manteve valores próximos a zero

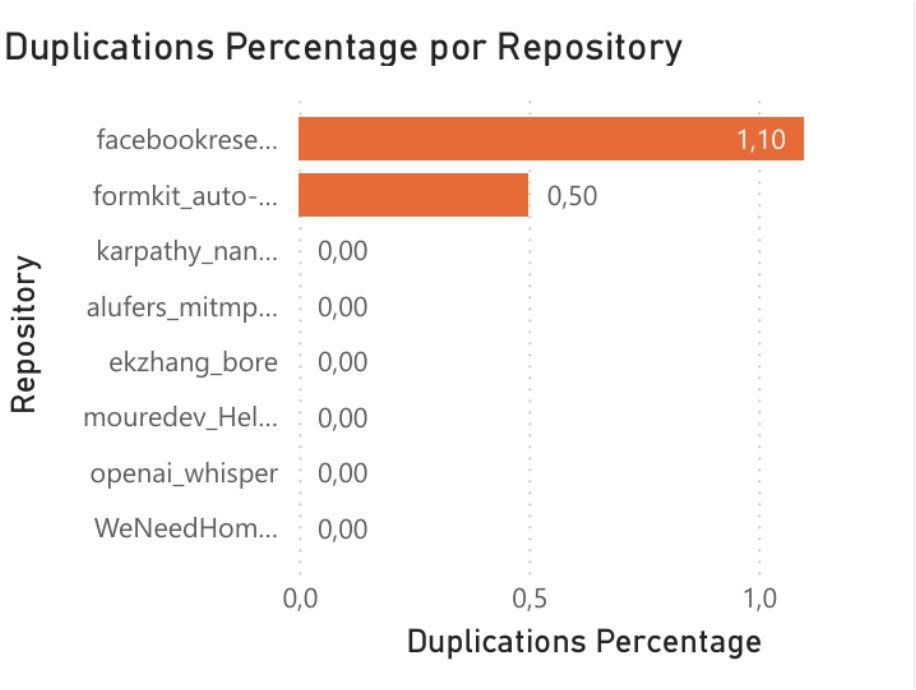

Figura 3: Percentual de duplicação de código por repositório nos períodos pré-morte e pós-revive

***Code Smells* e *Issues*:**

- ***Issues* Pré-morte**: Soma total de 10.479 dentre todos os projetos.
- ***Issues* Pós-revive**: Soma total de 24 novas dentre todos os projetos.
- ***Issues* Resolvidas Pós-revive**: Até 5 *issues* em `formkit_auto`
- **Novas *Issues* Pós-revive**: Até 21 *issues* em `formkit_auto`

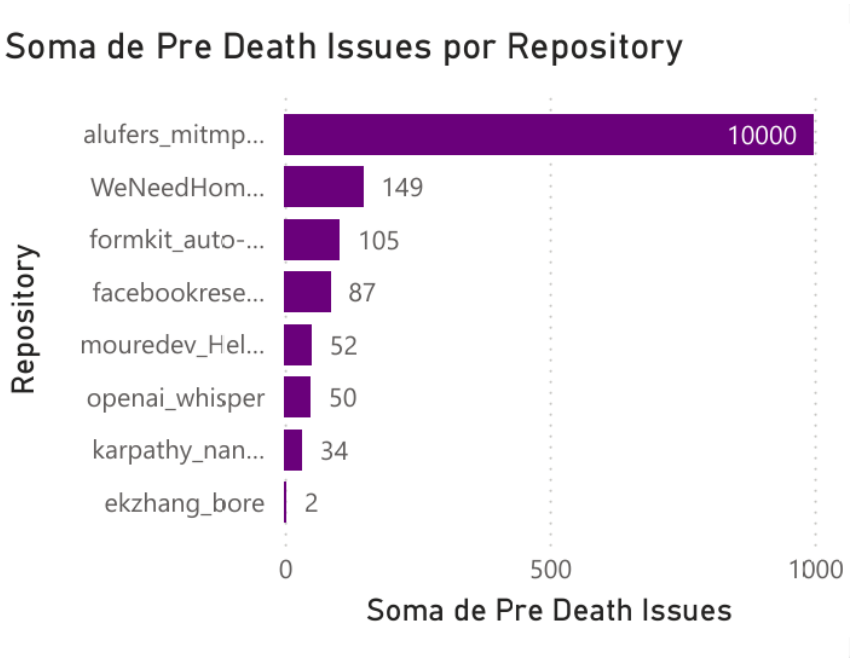

Figura 4: Quantidade de issues pré-morte por repositório

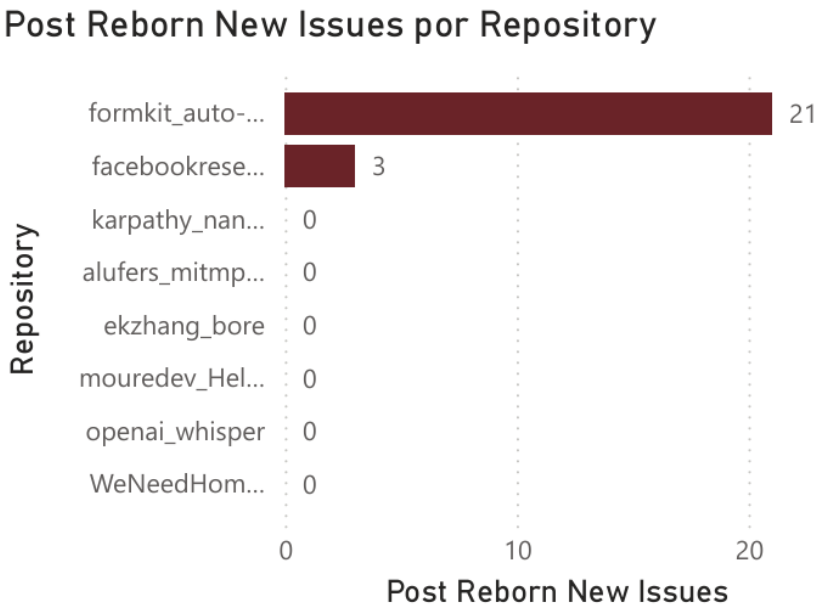

Figura 5: Issues resolvidas e novas issues no período pós-revive por repositório

Padrões Identificados:

- Projetos com maior atividade pós-revive tenderam a maiores mudanças
- Métricas de duplicação mostraram maior volatilidade
- *Maintainability score* demonstrou relativa estabilidade

4 Discussão

4.1 Interpretação dos Resultados

RQ1 - Evolução da Qualidade: Os resultados sugerem que a revivência não necessariamente implica em degradação ou melhoria sistemática da qualidade. A hipótese é parcialmente suportada, com diferenças significativas em métricas específicas (duplicação, *issues*) mas não em *scores* agregados. A heterogeneidade observada reflete diferentes estratégias de retomada de desenvolvimento.

RQ2 - Aspectos Mais Alterados: A duplicação de código e a quantidade de *issues* emergiram como as métricas mais sensíveis ao processo de revivência. A hipótese é suportada para estas métricas, enquanto *reliability* score mostrou menor variabilidade. Isto alinha-se com a literatura que identifica duplicação como métrica responsive a mudanças processuais.

4.2 Limitações do Estudo

Embora este estudo ofereça insights valiosos sobre a evolução da qualidade de código em repositórios revividos, é importante reconhecer suas limitações. Por se tratar de um estudo observacional, não é possível inferir causalidade direta entre a revivência dos repositórios e as mudanças observadas nas métricas de qualidade.

Ademais, o tamanho amostral modesto de apenas oito repositórios analisados limita a generalização dos achados para o universo mais amplo de projetos open-source. Outra limitação significativa refere-se à variabilidade inerente na definição operacional de "morte"e "revivência", o que introduz desafios metodológicos na comparação entre projetos. Somado a esses aspectos, a análise esteve intrinsicamente dependente das capacidades e limitações da ferramenta SonarQube, cujas métricas e thresholds podem não capturar completamente nuances específicas de qualidade em todos os contextos de desenvolvimento.

Por fim, cabe ressaltar o potencial viés de seleção na amostra, uma vez que os repositórios estudados tendem a ser mais populares e visíveis, podendo não representar adequadamente projetos menores ou menos conhecidos que também passaram por processos de revivência.

4.3 Trabalhos Futuros

Com base nas limitações identificadas e nos resultados obtidos, delineiam-se várias direções promissoras para pesquisas futuras. Em primeiro lugar, recomenda-se a ampliação da amostra com um maior número de repositórios ressuscitados, o que permitiria uma análise estatística mais robusta e generalizável.

Paralelamente, seria valioso conduzir um estudo longitudinal envolvendo múltiplos snapshots pós-revive, capturando assim a evolução temporal da qualidade do código além do momento inicial de retomada das atividades. Adicionalmente, a incorporação de métricas de processo - como frequência de commits, tamanho da equipe e padrões de colaboração - enriqueceria significativamente a compreensão dos fatores que influenciam a qualidade durante o processo de revivência.

Por fim, a realização de estudos qualitativos complementares, incluindo entrevistas com mantenedores dos projetos, poderia elucidar os contextos, motivações e desafios por trás dos processos de revivência, oferecendo assim uma perspectiva mais holística sobre este fenômeno complexo do ecossistema open-source.

4.4 Conclusão Geral

O estudo implementou um pipeline reprodutível para análise da qualidade de código em repositórios revividos, revelando padrões complexos e contextualmente dependentes. Enquanto alguns aspectos como duplicação de código mostraram sensibilidade ao processo de revivência, métricas agregadas como *maintainability* demonstraram notável resiliência. Estes achados contribuem para a compreensão das dinâmicas de qualidade em projetos *open-source* com históricos de inatividade.

Os resultados obtidos dialogam com as descobertas de Lenarduzzi et al. [2], que em um estudo em maior escala com 33 projetos Java identificaram que classes afetadas por itens de dívida técnica detectados pelo SonarQube apresentam efeito estatisticamente significativo, porém de pequena magnitude, na propensão a mudanças. Tal como observado em nossa pesquisa, os autores constataram que métricas de qualidade não necessariamente se degradam de forma acentuada em cenários de evolução de código, reforçando

a complexidade das relações entre características do processo de desenvolvimento e métricas de qualidade interna.

Ademais, o trabalho de Lenarduzzi et al. [2] evidencia incongruências na classificação de tipos e níveis de severidade dos itens de dívida técnica pelo SonarQube, o que ressalta a importância de interpretar com cautela os resultados das análises estáticas. Esta constatação corrobora a abordagem adotada em nosso estudo, que buscou analisar múltiplas dimensões da qualidade além dos *scores* agregados, considerando a natureza contextual das métricas em projetos revividos.

Em síntese, nossos achados, em consonância com a literatura recente, reforçam que a qualidade do código em projetos *open-source* é influenciada por fatores multidimensionais, sendo fundamental considerar tanto aspectos quantitativos quanto contextuais na avaliação de projetos que passaram por processos de revivência.

Referências

[1] SonarQube Documentation. *Static Code Analysis*. Disponível em: `https://docs.sonarqube.org/`

[2] LENARDUZZI, V.; SAARIMAKI, N.; TAIBI, D. **Some SonarQube Issues have a Significant but Small Effect on Faults and Changes. A large-scale empirical study**. 2025. Disponível em: `https://arxiv.org/abs/1908.11590v1`. Acesso em: 28/10/2025.