Pontifícia Universidade Católica de Minas Gerais
Laboratório de Experimentação de Software

RELATÓRIO

Uma Análise Comparativa de Repositórios Populares do GitHub: Linguagens, Contribuições e Frequência de Atualizações

A Comparative Analysis of Popular GitHub Repositories: Languages, Contributions, and Update Frequency

Lorrayne Oliveira [Pontifícia Universidade Católica de Minas Gerais | *lorrayne.marayze@gmail.com*]
Pedro Pires [Pontifícia Universidade Católica de Minas Gerais | *pedro.pires@gmail.com*]

Resumo. Este trabalho apresenta uma análise empírica abrangente das principais características dos 1.000 repositórios mais populares do GitHub, classificados de acordo com o número de estrelas atribuídas pelos usuários da plataforma. A popularidade medida por estrelas não apenas reflete a relevância de um projeto dentro da comunidade de desenvolvedores, mas também serve como indicador indireto de sua visibilidade, utilidade e adoção prática em diferentes contextos de desenvolvimento de software. Ao investigar sistematicamente esses repositórios, busca-se identificar padrões e características comuns que contribuem para o sucesso e longevidade de projetos open-source que alcançam ampla aceitação na comunidade global. O objetivo central desta pesquisa é compreender de forma estruturada os fatores que caracterizam sistemas de código aberto altamente populares, explorando suas dinâmicas de desenvolvimento, manutenção e evolução ao longo do tempo. A investigação foi conduzida através da coleta automatizada de dados utilizando a API GraphQL do GitHub, permitindo a extração eficiente e padronizada de informações sobre múltiplas dimensões dos repositórios analisados. Esta abordagem metodológica possibilitou a formação de um conjunto robusto e confiável de dados, fornecendo subsídios empíricos para avaliar aspectos fundamentais do desenvolvimento colaborativo. Foram formuladas e investigadas seis questões de pesquisa (RQs) fundamentais, cada uma associada a métricas específicas que mensuram diferentes facetas da dinâmica de projetos open-source bem-sucedidos, desde sua maturidade temporal até padrões de colaboração externa e práticas de gerenciamento.

Abstract. This work presents a comprehensive empirical analysis of the main characteristics of the 1,000 most popular GitHub repositories, ranked according to the number of stars assigned by users of the platform. Popularity measured by stars not only reflects the relevance of a project within the developer community, but also serves as an indirect indicator of its visibility, usefulness, and practical adoption in different software development contexts. By systematically investigating these repositories, this study seeks to identify common patterns and characteristics that contribute to the success and longevity of open-source projects that achieve wide acceptance in the global community. The central objective of this research is to systematically understand the factors that characterize highly popular open-source systems, exploring their dynamics of development, maintenance, and evolution over time. The investigation was conducted through the automated collection of data using GitHub's GraphQL API, which enabled the efficient and standardized extraction of information about multiple dimensions of the analyzed repositories. This methodological approach allowed the construction of a robust and reliable dataset, providing empirical evidence to assess fundamental aspects of collaborative development. Six fundamental research questions (RQs) were formulated and investigated, each associated with specific metrics that measure different facets of the dynamics of successful open-source projects, ranging from temporal maturity to patterns of external collaboration and management practices.

Palavras-chave: GitHub, repositórios populares, análise empírica, desenvolvimento open-source, métricas de software

Keywords: GitHub, popular repositories, empirical analysis, open-source development, software metrics

1 Introdução

O desenvolvimento de software open-source tem se tornado cada vez mais relevante no cenário tecnológico atual, com plataformas como o GitHub. A popularidade de um repositório, frequentemente medida pelo número de estrelas, pode ser um indicador de sua qualidade, utilidade e adoção pela comunidade. No entanto, existem questões importantes sobre quais características realmente definem um bom projeto de software no ambiente open-source.

Este estudo tem como objetivo investigar empiricamente as principais características dos repositórios mais populares do GitHub, analisando aspectos como maturidade, contribuição externa, frequência de releases, atualização, linguagens de programação e gerenciamento de issues. Através da coleta e análise de dados dos 1.000 repositórios com maior número de estrelas, busca-se compreender os padrões que caracterizam projetos de software de alta popularidade.

1.1 Hipóteses Informais

Com base na literatura científica existente e observações do ecossistema de desenvolvimento open-source, foram elaboradas as seguintes hipóteses informais para orientar a investigação:

H1 - Idade dos Repositórios: Espera-se que repositórios populares apresentem maior idade, geralmente superior a 5 anos, uma vez que projetos necessitam de tempo para ganhar reconhecimento, estabilidade e consolidar uma base de usuários. Estudos sobre evolução de software open-source

indicam que o fator temporal é determinante para o desenvolvimento de processos de qualidade e adoção pela comunidade Ait Houaich *et al*. [2015]; Badreddin *et al*. [2014], corroborando a expectativa de que popularidade está associada à longevidade dos projetos.

H2 - Contribuição Externa: Sistemas populares devem receber significativa contribuição externa, com centenas ou milhares de pull requests aceitas, refletindo o engajamento ativo da comunidade e a natureza colaborativa do desenvolvimento open-source. Pesquisas anteriores demonstram que a popularidade de repositórios GitHub está diretamente relacionada ao engajamento da comunidade Borges *et al*. [2016b], sugerindo uma correlação positiva entre estrelas e contribuições externas.

H3 - Frequência de Releases: Projetos populares provavelmente mantêm um ciclo regular de releases, com dezenas de versões lançadas, demonstrando evolução contínua e manutenção ativa. Estudos indicam que novos releases impactam significativamente a popularidade de projetos Borges *et al*. [2016a], suportando a hipótese de que repositórios populares mantêm atividade regular de lançamento.

H4 - Atualização Frequente: Espera-se que repositórios populares sejam atualizados com frequência, possivelmente com intervalos de dias a algumas semanas entre atualizações, indicando manutenção ativa e desenvolvimento contínuo. A literatura sugere que projetos ativos tendem a manter maior engajamento da comunidade e, consequentemente, maior popularidade Borges *et al*. [2016b].

H5 - Linguagens Populares: Sistemas populares devem ser predominantemente escritos em linguagens amplamente utilizadas como JavaScript, Python, Java, TypeScript e Go, refletindo tanto a popularidade dessas linguagens quanto sua adequação para projetos de grande escala. Pesquisas anteriores identificaram a linguagem de programação como um dos principais fatores que impactam a popularidade de repositórios GitHub Borges *et al*. [2016b].

H6 - Gerenciamento de Issues: Repositórios bem mantidos devem apresentar uma alta taxa de resolução de issues (acima de 70-80%), demonstrando responsividade da equipe de manutenção e qualidade do processo de gerenciamento de problemas. Estudos empíricos mostram que três em cada quatro desenvolvedores consideram o número de estrelas antes de adotar um projeto Borges *et al*. [2016b], sugerindo que a percepção de qualidade, refletida na gestão de issues, influencia a popularidade.

H7 – Popularidade da Linguagem: Repositórios desenvolvidos em linguagens mais populares tendem a atrair maior quantidade de contribuições externas, resultando em um ciclo de desenvolvimento mais dinâmico. Isso se reflete em maior número de releases publicados e atualizações mais frequentes. A escolha da linguagem pode impactar diretamente a visibilidade e a atratividade do projeto para novos colaboradores, funcionando como um facilitador de engajamento da comunidade.

1.2 Objetivos

O objetivo principal deste trabalho é caracterizar empiricamente os repositórios mais populares do GitHub através de métricas quantitativas, respondendo às seguintes questões de pesquisa:

- **RQ01:** Sistemas populares são maduros/antigos?

- **RQ02:** Sistemas populares recebem muita contribuição externa?

- **RQ03:** Sistemas populares lançam releases com frequência?

- **RQ04:** Sistemas populares são atualizados com frequência?

- **RQ05:** Sistemas populares são escritos nas linguagens mais populares?

- **RQ06:** Sistemas populares possuem um alto percentual de issues - fechadas?

- **RQ07:** Sistemas escritos em linguagens mais populares recebem mais contribuição externa, lançam mais releases e são atualizados com mais frequência?

Como objetivo secundário, pretende-se validar ou refutar as hipóteses informais elaboradas, contribuindo para o entendimento das características que definem o sucesso de projetos open-source.

2 Metodologia

2.1 Coleta de Dados

A coleta de dados foi realizada através da API GraphQL do GitHub, utilizando um script Python desenvolvido para esta análise. O processo de coleta seguiu os seguintes passos:

- **Autenticação:** Utilização de token de acesso pessoal do GitHub para autenticar as requisições à API.

- **Consulta:** Implementação de consulta GraphQL para buscar os repositórios com maior número de estrelas, utilizando a query ”stars:>1 sort:stars-desc”.

- **Paginação:** Implementação de sistema de paginação para coletar os 1.000 repositórios mais populares, utilizando 100 páginas com 10 repositórios cada.

- **Tratamento de Erros:** Implementação de mecanismos de tratamento de rate limiting e verificação de erros GraphQL.

2.2 Méticas Coletadas

Para cada repositório, foram coletadas as seguintes métricas:

- Nome e proprietário do repositório

- Data de criação (para cálculo da idade)

- Data da última atualização (para cálculo do tempo desde a última atualização)

- Número de estrelas (stargazerCount)

- Linguagem primária (primaryLanguage)

- Número total de releases

- Número total de pull requests aceitas (estado MERGED)

- Número total de issues (estados OPEN e CLOSED)

- Número de issues fechadas (estado CLOSED)

- Número de estrelas

- Número de contribuidores

2.3 Processamento dos Dados

Após a coleta, os dados brutos foram processados para calcular as métricas necessárias para responder às questões de pesquisa:

- **Idade do repositório:** Calculada em anos a partir da diferença entre a data atual e a data de criação.

- **Tempo desde última atualização:** Calculado em dias a partir da diferença entre a data atual e a data da última atualização.

- **Razão de issues fechadas:** Calculada como o número de issues fechadas dividido pelo número total de issues.

2.4 Exportação e Análise

Os dados processados foram exportados para um arquivo CSV (`repositorios_github_dados.csv`) com o objetivo de facilitar análises posteriores. A análise dos resultados será realizada utilizando medidas de tendência central, como a mediana, conforme especificado. Além disso, serão realizadas análises por categoria para variáveis categóricas, como a linguagem de programação.

2.5 Limitações

Este estudo possui algumas limitações que devem ser consideradas na análise dos resultados. Primeiramente, a popularidade é medida exclusivamente pelo número de estrelas do GitHub, o que pode não refletir completamente a qualidade, utilidade real ou impacto prático do software na comunidade. Embora as estrelas sejam um indicador amplamente aceito de popularidade Borges *et al.* [2016b], elas podem ser influenciadas por fatores externos como campanhas de marketing, tendências temporárias ou vieses relacionados à visibilidade de projetos de grandes organizações.

A classificação de linguagem primária utilizada pelo GitHub pode não refletir adequadamente a complexidade real de projetos multilíngues, especialmente em sistemas que combinam múltiplas tecnologias ou que possuem componentes significativos em diferentes linguagens. Por fim, as métricas coletadas representam valores absolutos que não são normalizados pelo tamanho, complexidade ou domínio de aplicação dos projetos, podendo favorecer certos tipos de repositórios em detrimento de outros.

3 Resultados

3.1 RQ01: Sistemas populares são maduros/antigos?

Existe uma concentração de repositórios populares em uma faixa intermediária de idade. A maior parte deles está entre 7 e 12 anos, com destaque para os de 9 a 10 anos, que chegam a 100 repositórios, sendo o maior quantidde. Esse período parece ser quando os projetos já alcançaram uma maturidade técnica e organizacional.

Projetos mais novos, com menos de 3 anos, aparecem em número bem menor. Isso sugere que a popularidade leva tempo para ser construída, já que nesses primeiros anos os repositórios ainda estão consolidando suas principais funcionalidades, documentação e práticas de desenvolvimento.

No outro extremo, os repositórios muito antigos, com mais de 15 anos, também são poucos. Isso pode estar ligado a problemas de obsolescência, maior dificuldade de manutenção ou até à substituição por tecnologias mais modernas. Como o ecossistema de software muda muito rápido, projetos que não acompanham essas mudanças acabam perdendo sua relevância.

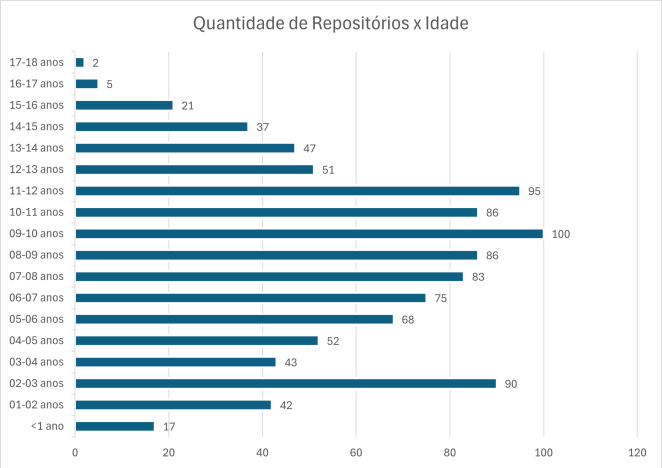

Figura 1. Quantidade de Repositórios x Idade

Os dados apoiam fortemente a hipótese inicial. A concentração de repositórios populares na faixa de 5-11 anos sugere que projetos precisam de tempo para amadurecer e construir sua reputação. Repositórios muito novos provavelmente ainda não tiveram tempo suficiente para desenvolver funcionalidades, documentação e uma comunidade ativa. Por outro lado, projetos muito antigos podem ter perdido relevância ou sido substituídos por tecnologias mais modernas.

A faixa de 7-12 anos representa um equilíbrio, onde os projetos já passaram pela fase inicial de instabilidade, desenvolveram funcionalidades essenciais, construíram uma base de usuários e ainda mantêm relevância tecnológica. Esse período também coincide com o tempo necessário para estabelecer processos de qualidade que tornam o projeto confiável para uso em produção.

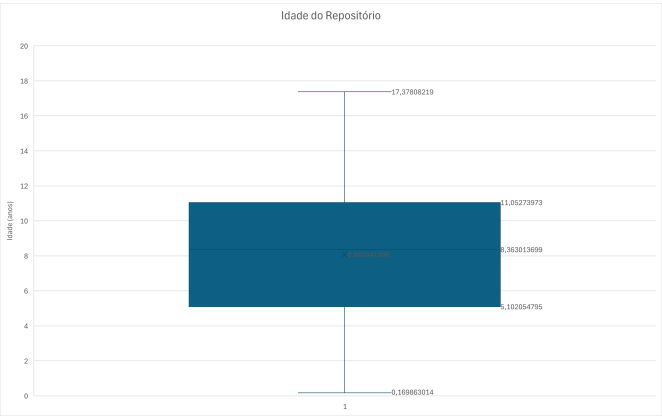

Figura 2. Boxplot Idade

A análise confirma que a popularidade em repositórios está diretamente relacionada à maturidade temporal. Projetos com idade superior a 5 anos têm maior probabilidade de serem populares, com o pico de popularidade ocorrendo entre 7 e 12 anos. Isso valida a hipótese de que o tempo é um fator crucial para que repositórios ganhem reconhecimento, estabilidade e consolidem uma base sólida de usuários na comunidade open-source.

Pudemos verificar que 76% dos repositórios tem mais que 5 anos desde a sua criação.

3.2 RQ02: Sistemas populares recebem muita contribuição externa?

O gráfico abaixo de Pull Requests x Idade mostra o volume acumulado de contribuições externas aceitas ao longo do tempo. Podemos observar que projetos com 8 a 12 anos de idade se destacam com o maior número de pull requests, alcançando picos de mais de 500 mil contribuições aceitas na faixa de 11 a 12 anos. Projetos mais novos (até 5 anos) apresentam volume de contribuições significativamente menor, o que pode estar associado ao tempo necessário para consolidar uma comunidade ativa de colaboradores. Por outro lado, projetos muito antigos (acima de 15 anos) também registram menor volume, possivelmente por já estarem mais estáveis ou terem reduzido o ritmo de evolução.

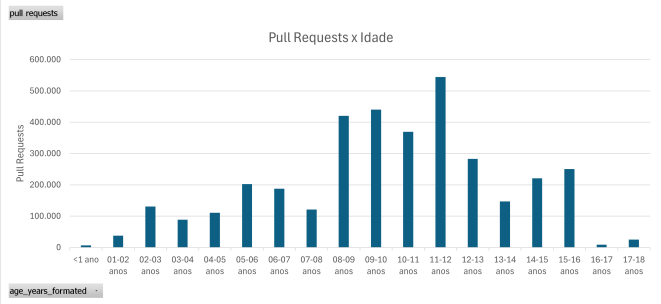

Figura 3. Pull Requests x Idade

No gráfico Contribuidores x Estrelas, observamos que a maior parte da popularidade (soma de estrelas) está concentrada em projetos com até 1000 contribuidores, acumulando mais de 217 mil estrelas. Isso indica que muitos sistemas populares não necessariamente dependem de um alto número de colaboradores para atingir grande visibilidade. Projetos com 1001 a 2000 contribuidores ainda concentram um volume expressivo (mais de 102 mil estrelas) e pareado a ele temos os projetos com 9001 a 10000 contribuidores que possuem mais de 79 mil estrelas. Mas nas demais faixas de contribuidores há uma queda significativa. Isso sugere que a relação entre popularidade e número de contribuidores é desigual: nem sempre os projetos mais estrelados contam com uma grande comunidade, mas sim com uma base moderada de contribuidores ativos.

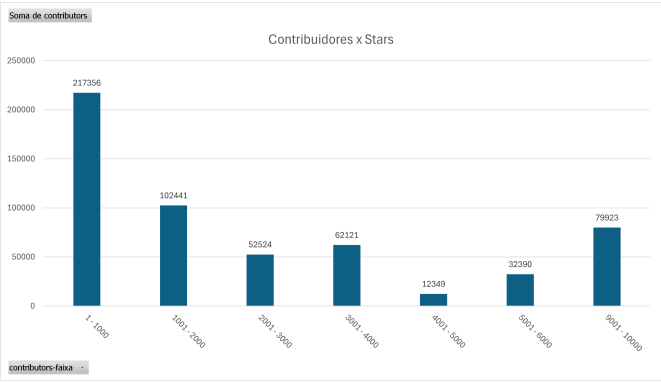

Figura 4. Contribuidores x Estrelas

Os resultados sugerem que sistemas populares de fato recebem contribuições externas em grande volume, mas esse volume não depende apenas da quantidade de contribuidores, e sim da maturidade e relevância do projeto. Projetos em fase madura (8–12 anos) conseguem equilibrar comunidade ativa e estabilidade, gerando grandes quantidades de pull requests. Além disso, observa-se que a maior parte da popularidade está concentrada em projetos com até 1000 contribuidores, o que mostra que não é necessário ter uma comunidade grande para manter alto nível de participação externa.

3.3 RQ03: Sistemas populares lançam releases com frequência?

Observa-se que a maior parte das releases está concentrada em repositórios com 27.500 a 50.000 estrelas, acumulando mais de 68 mil releases. Em seguida, repositórios com 50.001 a 100.000 estrelas concentram mais de 32 mil releases, e aqueles na faixa de 100.001 a 150.000 estrelas somam pouco mais de 7 mil releases. A partir desse ponto, há uma queda acentuada: quase não existem repositórios acima de 150 mil estrelas com grande volume de releases. Isso sugere que, embora popularidade (medida pelo número de estrelas) esteja associada a maior quantidade de releases, essa relação não é linear. Na prática, poucos projetos extremamente populares ultrapassam a marca de 150 mil estrelas, e muitos deles podem ter estratégias diferentes de versionamento (por exemplo, concentrando mudanças em releases maiores e menos frequentes).

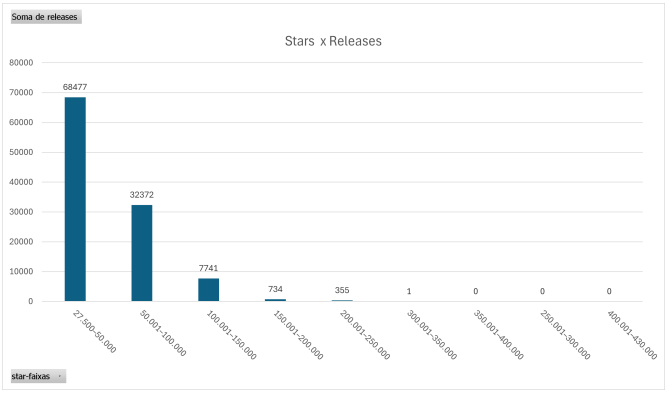

Figura 5. Realeases x Estrelas

Já no gráfico Releases x Idade, a distribuição mostra um padrão mais gradual. Projetos entre 9 e 11 anos de idade concentram o maior volume de releases, com destaque para os de 9 anos, que ultrapassam 12 mil releases. Projetos na faixa de 10 a 11 anos mantêm valores elevados (acima de 9 mil releases), enquanto projetos mais novos (menos de 5 anos) ou mais antigos (acima de 15 anos) apresentam quantidades bem menores. Isso sugere que existe um ciclo de maturidade, onde sistemas jovens ainda estão em fase de consolidação e não acumulam tantas versões, sistemas maduros (9–11 anos) estão em seu auge de atividade, com releases frequentes e contínuas e sistemas muito antigos podem apresentar uma queda no ritmo de releases, possivelmente devido à estabilização da base de código ou ao declínio da comunidade de manutenção.

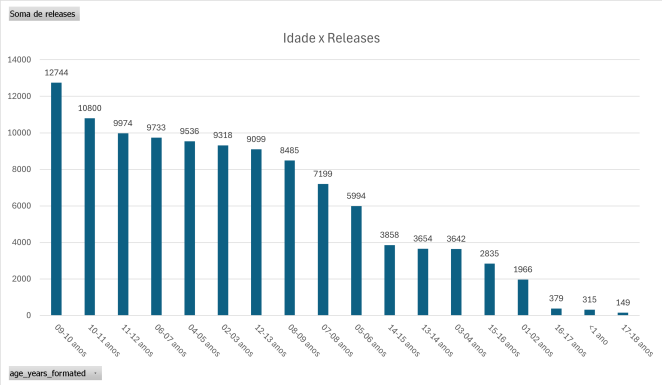

Figura 6. Realeases x Idade

3.4 RQ04: Sistemas populares são atualizados com frequência?

3.5 RQ05: Sistemas populares são escritos nas linguagens mais populares?

3.6 RQ06: Sistemas populares possuem um alto percentual de issues fechadas?

3.7 RQ07: Sistemas escritos em linguagens mais populares recebem mais contribuição externa, lançam mais releases e são atualizados com mais frequência?

4 Conclusão

Referências

Ait Houaich, Y. *et al*. (2015). Measuring the maturity of open source software. *ResearchGate*. Preprint.

Badreddin, O., Keshta, I., and Bekhet, S. (2014). A study on maturity model of open source software community to estimate the quality of products. *Procedia Computer Science*, 35:1282–1291. DOI: 10.1016/j.procs.2014.08.233.

Borges, H., Hora, A., and Valente, M. T. (2016a). Predicting the popularity of github repositories. In *Proceedings of the 12th International Conference on Predictive Models and Data Analytics in Software Engineering*, pages 1–10. ACM. DOI: 10.1145/2972958.2972966.

Borges, H., Hora, A., and Valente, M. T. (2016b). Understanding the factors that impact the popularity of github repositories. In *Proceedings of the 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, pages 334–344. IEEE. DOI: 10.1109/SANER.2016.31.