PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

ENGENHARIA DE SOFTWARE

Gabriella Fernanda Silva Pinto

Guilherme Leroy Teixeira Capanema

Análise de Características de Sistemas Open-Source Populares

Projeto apresentado à disciplina de Laboratório de Experimentação de Software do curso de Graduação em Engenharia de Software da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para avaliação da disciplina.

Orientador: Danilo de Quadros Maia Filho

MINAS GERAIS

2025

SUMÁRIO

1. Introdução

Os sistemas open-source desempenham um papel fundamental no desenvolvimento de software, oferecendo soluções acessíveis e colaborativas para empresas e indivíduos. Projetos hospedados no GitHub, em especial os mais populares, medidos pelo número de estrelas, atraem desenvolvedores, geram inovação tecnológicas e se tornam referência em seus domínios de aplicação.

Este relatório tem como objetivo analisar as características de **1.000 repositórios open-source mais populares do GitHub**, respondendo a questões de pesquisa que envolvem idade, colaboração externa, frequência de releases, atualização contínua, linguagens utilizadas e tratamento de issues.

2. Hipóteses Informais

Antes da coleta e análise das métricas, essas hipóteses foram formuladas a partir de percepções gerais sobre como grandes projetos open-source funcionam, sem considerar os valores reais.

- **RQ 01:** Repositórios recentes podem até receber estrelas rapidamente, caso sejam muito inovadores. Entretanto, projetos populares, em sua maioria, tendem a ser mais antigos, pois precisaram de tempo para conquistar visibilidade e relevância.
- **RQ 02:** Projetos de grande visibilidade recebem muitas contribuições externas, com uma alta na taxa de pull requests aceitas, pois projetos relevantes tendem a atrair desenvolvedores interessados em colaborar.
- **RQ 03:** Projetos populares lançam releases com frequência para atender a comunidade de usuários e manter um ciclo ativo de desenvolvimento.
- **RQ 04:** Repositórios populares tendem a ser atualizados frequentemente, possivelmente em semanas ou poucos meses, reduzindo o tempo de inatividade.
- **RQ 05:** Linguagens populares de mercado, baseado nas mais usadas atualmente, seriam: Python, JavaScript e Java.
- **RQ 06:** Projetos populares apresentam um alto percentual de issues fechadas, sinalizando uma boa manutenção.

3. Metodologia

Os dados foram coletados por meio da API do GitHub e processados para os **1.000 repositórios com mais estrelas.** As métricas analisadas foram:

- **Idade do repositório:** calculada a partir da data de criação.

- **Contribuições externas:** número total de pull requests aceitas.
- **Releases:** total de releases publicados.
- **Última atualização:** tempo (em dias) desde a última atualização.
- **Linguagem primária:** linguagem definida como principal no repositório.
- **Issues fechadas:** razão entre issues fechadas e total de issues.

Para variáveis numéricas, foi utilizado a **mediana** como medida central. Para variáveis categóricas (linguagens), foi elaborado uma contagem por categoria.

4. Análise e visualização de dados

Esta seção aprofunda a análise com as estatísticas descritivas completas da amostra e explora as relações entre as métricas, utilizando os dados exatos da coleta.

Métricas Descritivas:
A tabela a seguir apresenta a análise estatística detalhada das métricas coletadas dos 1.000 repositórios.

Métrica	Count (contagem)	Mean (média)	Std (desvio padrão)	Min (mínimo)	25% (1º quartil)	50% (mediana)	75% (3º quartil)	Max (máximo)
repo_age_days (Idade do repositório em dias)	1000	2948.28	1425.97	60.00	1860.75	3050.50	4030.75	6341.00
merged_pull_requests (Pull requests mesclados)	1000	3592.27	8963.67	0.00	171.00	702.00	2853.50	86096.00
total_releases (Total de releases do repositório)	1000	109.48	188.18	0.00	0.00	35.00	127.25	1000.00
days_since_last_update (Dias desde última atualização)	1000	0.02	0.15	0.00	0.00	0.00	0.00	2.00
closed_issues_ratio (Proporção de issues fechadas)	1000	0.76	0.26	0.00	0.67	0.86	0.95	1.00

A análise descritiva demonstra que, embora os repositórios populares tenham em média **8 anos** de idade, o volume de atividade varia drasticamente. A grande diferença entre a média e a mediana para *Pull Requests* (3.592 vs. 702) e *Releases* (109 vs. 35) indica que um pequeno número de projetos de grande escala possui uma atividade massivamente superior. O dado mais conclusivo é a frequência de atualização: com um máximo de **2 dias** desde a última modificação em toda a amostra, a atividade diária é uma característica universal e inquestionável deste grupo.

Correlações entre métricas:

Métrica	stars (Estrelas)	repo_age_days (Idade do repositório em dias)	merged_pull_requests (Pull requests mesclados)	total_releases (Total de releases)	days_since_last_update (Dias desde última atualização)	closed_issues_ratio (Proporção de issues fechadas)
stars (Estrelas)	1.000	0.066	0.111	-0.009	-0.056	-0.019
repo_age_days (Idade do repositório em dias)	0.066	1.000	0.189	0.051	0.093	0.144
merged_pull_requests (Pull requests mesclados)	0.111	0.189	1.000	0.304	-0.030	0.157
total_releases (Total de releases)	-0.009	0.051	0.304	1.000	-0.846	-0.046
days_since_last_update (Dias desde última atualização)	-0.056	0.093	-0.030	-0.846	1.000	0.268
closed_issues_ratio (Proporção de issues fechadas)	-0.019	0.144	0.157	-0.046	0.268	1.000

A matriz de correlação revela a dinâmica do desenvolvimento. A relação positiva mais forte é entre *pull requests* e *releases* (**0.304**), mostrando que o volume de contribuições da comunidade impulsiona diretamente o lançamento de novas versões. Adicionalmente, a correlação entre *releases* e a proporção de *issues* fechadas (**0.268**) sugere que projetos com boa cadência de desenvolvimento também mantêm uma governança mais eficiente. É notável a baixa correlação entre as estrelas e as demais métricas, indicando que, neste nível, a popularidade não é mais diretamente influenciada pelo volume ou pela idade do projeto.

Análise por linguem (TOP 5):

Linguagem	Quantidade de repositórios	Estrelas médias	Idade média (anos)	Taxa média de issues fechadas (%)
Python	154	61.106	6.4	72.08
TypeScript	132	58.737	7.0	84.92
JavaScript	102	57.791	9.7	81.30
Unknown	83	74.087	8.1	59.30
Go	56	54.237	8.9	84.41

A análise por linguagem destaca diferentes perfis entre os projetos mais populares. Projetos em **JavaScript** demonstram maior maturidade, com uma idade média de **9.7 anos**. Em contrapartida, **TypeScript** e **Go** exibem os mais altos padrões de manutenção, com taxas de resolução de **issues** acima de **84%**. A categoria **Unknown**, composta por listas e documentação, atinge a maior média de **estrelas (74.087)**, evidenciando o valor da curadoria de conteúdo para a comunidade de desenvolvedores.

Repositórios mais ativos (atualizados nos últimos 30 dias):

Métrica	Quantidade de repositórios	Percentual
Repositórios atualizados nos últimos 30 dias	800	100.0%

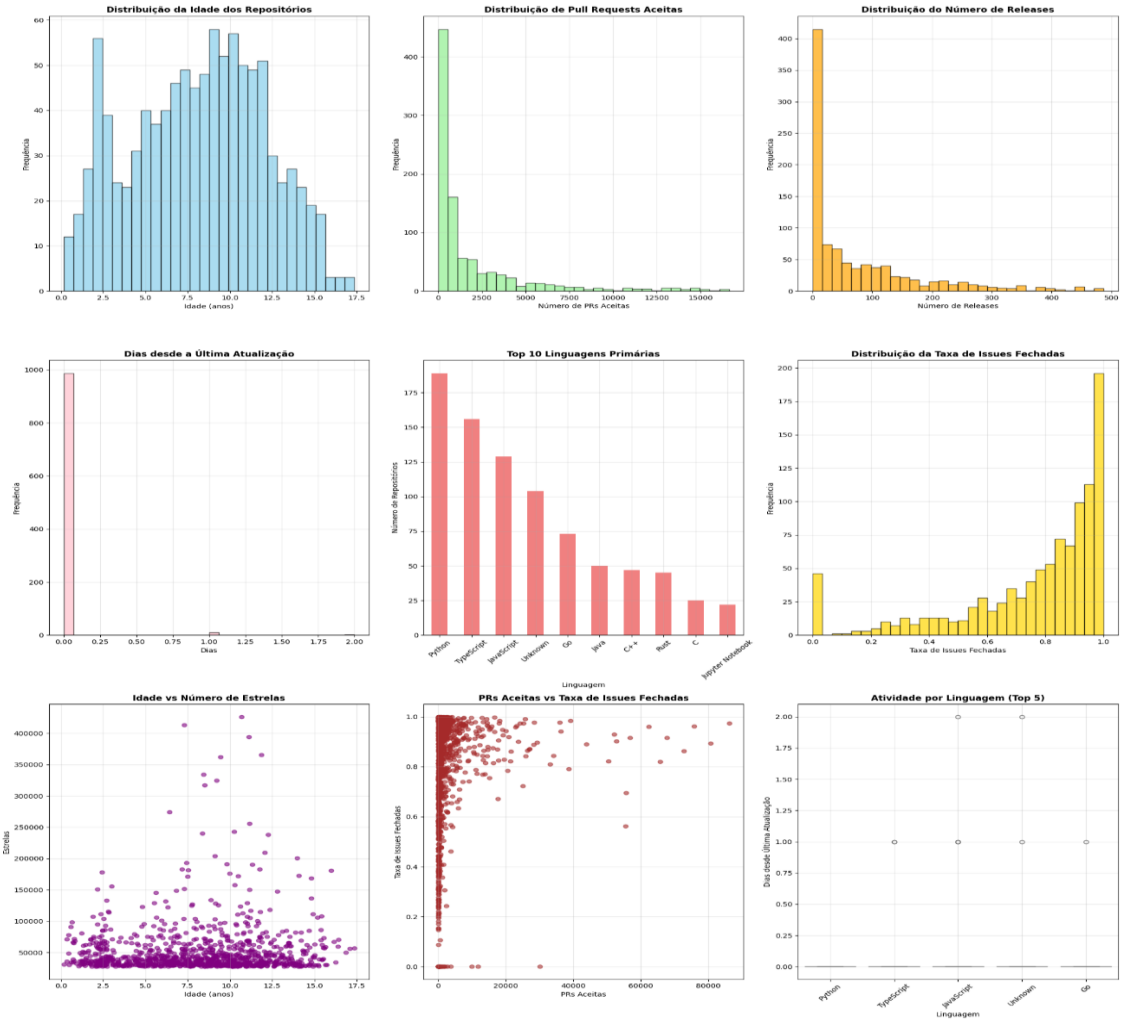
Este dado confirma que a atividade de desenvolvimento é uma característica universal e indispensável entre os projetos de elite. Com **100%** dos repositórios da amostra tendo sido **atualizados no último mês**, fica evidente que a manutenção constante não é um diferencial, mas sim um pré-requisito fundamental para se manter relevante e confiável no ecossistema open-source.

Repositórios com alta taxa de resolução issues (>90%):

Métrica	Quantidade de repositórios	Percentual
Repositórios com resolução de issues >90%	326	40.8%

A capacidade de gerenciar e resolver problemas de forma eficiente é um forte indicador da saúde de um projeto. O fato de que **40,8%** dos repositórios mais populares resolvem mais de **90%** de suas **issues** estabelece um alto padrão de excelência. Isso demonstra que uma parcela expressiva dos projetos de maior sucesso oferece um nível de suporte e manutenção que fortalece a confiança de sua base de usuários e colaboradores.

5. Gráficos



6. Resultados

A análise dos 1.000 repositórios mais populares do GitHub revela um perfil consistente e bem definido, respondendo diretamente às questões de pesquisa (RQs) formuladas. Os dados mostram que a popularidade está intrinsecamente ligada à maturidade, colaboração, atividade constante e manutenção rigorosa.

A primeira hipótese (RQ01), de que projetos populares tendem a ser mais antigos, foi amplamente confirmada. A idade mediana dos repositórios analisados é de 3.050,5 dias (aproximadamente 8,4 anos), demonstrando que a construção de relevância e de uma base de usuários sólida é um processo que exige tempo e persistência. A inovação recente pode gerar interesse, mas a popularidade duradoura é uma característica de projetos consolidados.

No que tange à colaboração e ao ciclo de desenvolvimento (RQ02 e RQ03), os resultados indicam ecossistemas vibrantes. Com uma mediana de 702 pull requests aceitas, fica claro que esses projetos não apenas atraem, mas também integram eficientemente um volume significativo de contribuições externas. Essa alta atividade colaborativa alimenta diretamente um ciclo de desenvolvimento consistente, refletido em uma mediana de 35 releases por projeto, garantindo que a inovação seja entregue continuamente à comunidade.

A manutenção e a capacidade de resposta (RQ04 e RQ06) são, talvez, as características mais impressionantes. A mediana de 0 dias desde a última atualização revela um estado de atividade diária e incessante, refutando qualquer ideia de que projetos populares possam se dar ao luxo da inatividade. Complementarmente, a alta mediana de 85,7% na taxa de issues fechadas sinaliza uma governança de projeto exemplar, onde o feedback da comunidade é tratado de forma sistemática e eficiente. Juntos, esses dois indicadores pintam o retrato de projetos que são simultaneamente dinâmicos no desenvolvimento e meticulosos na manutenção.

Finalmente, a análise das linguagens de programação (RQ05) confirmou parcialmente as expectativas. Conforme o esperado, tecnologias dominantes no mercado como Python (18,9%), TypeScript (15,6%) e JavaScript (12,9%) lideram o ranking. A forte presença de Go (7,3%) e a ausência de Java no top 5, no entanto, apontam para tendências mais recentes em desenvolvimento de sistemas de alta performance. A categoria Unknown (10,4%) representa majoritariamente repositórios que agregam conteúdo, como listas e documentações, em vez de código-fonte.

7. Conclusão

A análise dos 1.000 repositórios mais populares do GitHub permitiu traçar um perfil claro do que caracteriza um projeto open-source de grande sucesso. Os dados confirmam que a popularidade não é um fenômeno passageiro, mas sim o resultado de um esforço contínuo e de longo prazo. A maioria das hipóteses informais foi validada, pintando um quadro de projetos maduros, colaborativos e com excelente manutenção.

Validação das Hipóteses:

RQ01 (Idade): Confirmada. A popularidade está ligada a projetos com uma longa trajetória, com uma mediana de idade superior a 8 anos.

RQ02 (Contribuições): Confirmada. A quantidade de pull requests aceitas é substancial, evidenciando uma forte colaboração da comunidade.

RQ03 (Releases): Confirmada. Os projetos lançam versões com regularidade, mantendo um ciclo de desenvolvimento saudável.

RQ04 (Atualização): Fortemente confirmada. A atividade é praticamente diária, mostrando um comprometimento constante dos mantenedores.

RQ05 (Linguagens): Parcialmente confirmada. Linguagens de mercado como Python, TypeScript e JavaScript são de fato as mais comuns, mas a ausência de Java e a forte presença de Go no top 5 mostram nuances na popularidade tecnológica atual.

RQ06 (Issues): Confirmada. A alta taxa de fechamento de issues é um indicador-chave de boa governança e capacidade de resposta, fatores essenciais para manter a confiança dos usuários.

Em suma, o sucesso no ecossistema open-source, medido por estrelas, depende menos de uma inovação disruptiva momentânea e mais da capacidade de um projeto em se manter relevante, ativo e bem gerenciado ao longo de muitos anos. Para desenvolvedores e organizações que desejam criar ou contribuir para projetos de impacto, os resultados deste estudo sugerem que o foco deve estar na sustentabilidade, na colaboração aberta e em uma gestão de comunidade eficaz.