Análise de Engajamento e Colaboração em Repositórios Open Source no GitHub

Uma Investigação Sobre Padrões de Contribuição e Interação Comunitária

Pedro Araújo Franco
André Almeida
Renato Cazzoletti
Gabriel Ferreira Amaral
Davi Aguilar
Felipe Picinin

Instituto de Ciências Exatas e Informática (ICEI)
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
Belo Horizonte – MG – Brasil

21 de novembro de 2025

Resumo

Este estudo investiga os padrões de engajamento e colaboração em repositórios open source hospedados no GitHub, analisando a relação entre popularidade, contribuição ativa e interação comunitária. Apesar da importância crescente do desenvolvimento colaborativo de software, pouco se compreende sobre os fatores que influenciam o nível de participação e a qualidade das interações em projetos de código aberto. A pesquisa examina aspectos como a correlação entre estrelas e número de contribuidores, o nível de atividade da comunidade através de pull requests e commits, e a dinâmica de interação medida pelo tempo de resposta e gestão de issues. Utilizando o modelo GQM (Goal Question Metrics), o estudo propõe três questões de pesquisa que orientam a investigação sobre os elementos que caracterizam repositórios bem-sucedidos e comunidades engajadas. Os resultados podem contribuir para desenvolvedores que buscam aumentar a participação em seus projetos, mantenedores de repositórios, e pesquisadores interessados em dinâmicas de colaboração em plataformas de desenvolvimento social.

1 Introdução

O GitHub é atualmente a maior plataforma de hospedagem e colaboração de código-fonte do mundo, reunindo milhões de desenvolvedores e bilhões de linhas de código em projetos open source. Com mais de 100 milhões de usuários ativos e repositórios que abrangem praticamente todas as áreas da engenharia de software, a plataforma tornou-se um ecossistema fundamental para o desenvolvimento colaborativo moderno. Nesse contexto, os

repositórios open source não são apenas locais de armazenamento de código, mas verdadeiras comunidades onde desenvolvedores compartilham conhecimento, resolvem problemas complexos e contribuem coletivamente para a evolução tecnológica.

Apesar da popularidade e do impacto do GitHub no desenvolvimento de software, existe uma lacuna significativa na compreensão dos fatores que determinam o sucesso de um projeto open source em termos de engajamento comunitário. Enquanto métricas superficiais como o número de estrelas frequentemente são usadas como indicadores de popularidade, pouco se sabe sobre como esses indicadores se relacionam com a participação efetiva da comunidade, a qualidade das contribuições e a sustentabilidade de longo prazo dos projetos. Esta falta de compreensão sistemática limita a capacidade dos mantenedores de projetos de otimizar suas estratégias de engajamento e dificulta que novos projetos estabeleçam comunidades ativas e produtivas.

O problema específico que este trabalho busca abordar é a análise empírica das relações entre diferentes métricas de engajamento em repositórios open source. Especificamente, investigamos como a popularidade (medida por estrelas) se relaciona com o tamanho e a atividade da comunidade de contribuidores, como se manifesta a contribuição ativa através de pull requests e commits de desenvolvedores externos, e como ocorre a interação entre mantenedores e a comunidade através do gerenciamento de issues. Este recorte permite uma análise focada e quantificável dos aspectos mais críticos do engajamento comunitário em projetos de código aberto.

A motivação para realizar este estudo deriva da importância crescente do modelo de desenvolvimento open source na indústria de software contemporânea. Grandes empresas de tecnologia, startups e desenvolvedores independentes dependem cada vez mais de projetos de código aberto, e o sucesso desses projetos está intrinsecamente ligado à saúde e ao engajamento de suas comunidades. Compreender os padrões que caracterizam repositórios bem-sucedidos pode auxiliar desenvolvedores a criar e manter projetos mais atrativos, ajudar contribuidores a identificar projetos com comunidades ativas, e informar a criação de ferramentas que promovam colaboração mais efetiva.

A resolução deste problema é importante por diversas razões práticas e teóricas. Do ponto de vista prático, desenvolvedores que compreendem os fatores que influenciam o engajamento podem aplicar estratégias mais eficazes para atrair e reter contribuidores, aumentando assim a sustentabilidade de seus projetos. Para a comunidade acadêmica, este estudo contribui para o corpo crescente de pesquisas sobre engenharia de software social, fornecendo dados empíricos sobre as dinâmicas de colaboração em plataformas digitais. Além disso, os insights obtidos podem informar o desenvolvimento de sistemas de recomendação no GitHub e outras plataformas similares, melhorando a experiência dos usuários ao conectar desenvolvedores com projetos alinhados aos seus interesses e níveis de habilidade.

O objetivo geral deste trabalho é analisar os padrões de engajamento e colaboração em repositórios open source do GitHub, identificando relações entre métricas de popularidade, contribuição ativa e interação comunitária. Como objetivos específicos, pretendemos: (1) investigar a correlação entre o número de estrelas e o tamanho da comunidade de contribuidores; (2) avaliar o nível de participação ativa da comunidade através de pull requests e commits de contribuidores externos; e (3) examinar a qualidade da interação entre mantenedores e comunidade através do gerenciamento de issues e tempo de resposta.

Para atingir esses objetivos, este estudo utiliza o modelo GQM (Goal Question Metrics) para estruturar três questões de pesquisa principais. A primeira questão (Q1) investiga qual a relação entre o número de estrelas de um repositório e sua quantidade de con-

tribuidores, buscando entender se repositórios mais populares efetivamente atraem mais colaboradores. A segunda questão (Q2) examina qual é o nível de contribuição ativa da comunidade do projeto, analisando pull requests e commits de desenvolvedores externos para mensurar o envolvimento real além de métricas superficiais. A terceira questão (Q3) explora como ocorre a interação da comunidade em torno do repositório, investigando a responsividade dos mantenedores e o engajamento em reportar e resolver problemas através de issues.

A estrutura deste artigo está organizada para explorar de forma sistemática as dinâmicas de engajamento em repositórios open source do GitHub. Após esta introdução, a Seção 2 apresenta os trabalhos relacionados, contextualizando esta pesquisa no panorama acadêmico existente e destacando como este estudo complementa e estende investigações anteriores. A Seção 3 detalha a metodologia empregada, explicando as etapas de coleta de dados através da API do GitHub, processamento das informações, cálculo das métricas e análise dos resultados. A Seção 4 apresenta os resultados obtidos, com visualizações e análises quantitativas que respondem às três questões de pesquisa levantadas. Finalmente, a Seção 5 oferece as conclusões do estudo, sintetizando os principais achados, discutindo as limitações da pesquisa, e sugerindo direções para trabalhos futuros que possam aprofundar a compreensão sobre colaboração em plataformas de desenvolvimento social.

2 Trabalhos Relacionados

Esta seção apresenta estudos que contribuem para a compreensão dos fatores que influenciam o engajamento e a colaboração em repositórios open source, abordando métricas de popularidade, padrões de contribuição e dinâmicas comunitárias em plataformas de desenvolvimento colaborativo.

2.1 Métricas de Popularidade e Visibilidade

A popularidade de repositórios no GitHub tem sido objeto de investigação sistemática. Borges et al. (2016) [1] conduziram um estudo abrangente sobre os fatores que impactam a popularidade de projetos no GitHub, estabelecendo o número de estrelas como principal indicador de reconhecimento social e visibilidade na plataforma. Os autores analisaram 200 mil repositórios e identificaram que características como idade do projeto, linguagem de programação e tamanho da equipe são preditores significativos de popularidade. Este trabalho fornece a base conceitual para a utilização de estrelas como métrica de popularidade em nossa primeira questão de pesquisa.

Complementando essa perspectiva, Tov et al. (2018) [2] investigaram as práticas de atribuição de estrelas no GitHub, revelando que este mecanismo funciona como indicador de interesse e reconhecimento social, mas não necessariamente reflete contribuição ativa ou engajamento direto. O estudo demonstrou que aproximadamente 30% dos usuários que marcam um repositório com estrela nunca interagem ativamente com o projeto. Esta distinção entre popularidade e participação efetiva é fundamental para interpretar os resultados da Q1 deste trabalho, que explora justamente a relação entre visibilidade (estrelas) e tamanho da comunidade ativa (contribuidores).

2.2 Caracterização de Repositórios Ativos

A definição de critérios objetivos para identificar repositórios ativos é essencial para garantir a validade de estudos empíricos. Arachchi e Perera (2018) [3], em seu trabalho "Uncovering the Hidden Patterns of Contributor Engagements in Active and Inactive GitHub Projects", propuseram um conjunto robusto de métricas para distinguir projetos ativos de inativos. Os autores analisaram mais de 10 mil repositórios e estabeleceram oito critérios quantitativos: pelo menos 5 contribuidores, mais de 1 issue aberta, mais de 50 issues fechadas, mais de 100 estrelas, mais de 10 forks, mais de 50 commits, e último commit ocorrido nos últimos 180 dias. Esta metodologia demonstrou 94% de precisão na identificação de projetos genuinamente ativos.

Este trabalho é particularmente relevante para nossa pesquisa, pois fornece a base metodológica para a seleção da amostra. Ao adotar os mesmos critérios, garantimos que os 1000 repositórios analisados representam projetos com comunidades ativas e engajadas, eliminando projetos abandonados ou com atividade esporádica que poderiam distorcer as análises de engajamento e colaboração. Além disso, Arachchi e Perera identificaram padrões distintos de comportamento entre projetos ativos e inativos, observando que projetos ativos apresentam maior consistência nas métricas de interação comunitária, achado que dialoga diretamente com nossa Q3 sobre dinâmicas de interação.

2.3 Contribuições Externas e Participação Comunitária

A caracterização de contribuidores externos e a análise de seus padrões de participação são aspectos cruciais para compreender a vitalidade de comunidades open source. Padhye et al. (2014) [4], em "A Study of External Community Contribution to Open-Source Projects on GitHub", realizaram uma investigação pioneira sobre o papel de contribuidores externos (definidos como desenvolvedores que não pertencem à equipe central do projeto) na evolução de projetos open source. Os autores analisaram 100 repositórios populares e demonstraram que contribuidores externos são responsáveis por aproximadamente 25% dos commits totais, mas contribuem de forma desproporcional para a correção de bugs (39%) em comparação com novas funcionalidades (18%).

Este estudo introduziu uma metodologia sistemática para identificar e categorizar contribuidores externos, diferenciando-os de membros principais da equipe através de critérios como frequência de contribuição, permissões de escrita no repositório e afiliação organizacional. A distinção é operacionalizada verificando se o contribuidor é membro da organização oficial do repositório ou se possui histórico consistente de commits ao longo do tempo. Esta abordagem é fundamental para nossa Q2, que avalia o nível de contribuição ativa da comunidade através de pull requests e commits de desenvolvedores externos.

Padhye e colaboradores também observaram que projetos com maior proporção de contribuições externas tendem a apresentar maior diversidade de perspectivas e soluções, mas requerem processos de revisão mais estruturados. Esse achado complementa nossa investigação ao sugerir que o volume de contribuições externas, medido na Q2, deve ser interpretado em conjunto com as métricas de interação e resposta da Q3 para uma compreensão holística do engajamento comunitário.

2.4 Dinâmicas de Interação e Resposta

O tempo de resposta e a gestão de issues são indicadores importantes da saúde de comunidades open source. Yu et al. (2024) [5] investigaram os fatores que influenciam

a latência de primeira resposta em pull requests, demonstrando que o tempo médio de resposta de mantenedores varia significativamente entre projetos (de horas a semanas) e está correlacionado com a taxa de retenção de contribuidores. Projetos com tempo médio de resposta inferior a 48 horas apresentam 3,2 vezes mais probabilidade de reter contribuidores ocasionais.

Rahman e Roy (2014) [6] exploraram empiricamente o uso de templates de issues em projetos de larga escala, identificando que repositórios com templates estruturados apresentam issues mais completas e tempos de resolução 30% menores. Esses achados contextualizam nossa métrica M1 da Q3, que mensura o tempo de resposta às issues como indicador de engajamento dos mantenedores com a comunidade.

2.5 Lacunas e Contribuições Deste Estudo

Embora os trabalhos anteriores tenham estabelecido fundamentos importantes, observam-se lacunas significativas na literatura. Poucos estudos investigam de forma integrada a relação entre popularidade (estrelas), contribuição efetiva (commits e pull requests de externos) e responsividade (gestão de issues), componentes essenciais para caracterizar comunidades engajadas. Além disso, a maioria das pesquisas foca em análises isoladas de métricas específicas, sem examinar suas inter-relações.

Este trabalho contribui ao preencher essas lacunas através de uma análise sistemática e integrada de 1000 repositórios ativos, investigando simultaneamente: (1) se a popularidade prediz o tamanho da comunidade de contribuidores; (2) qual o nível real de participação ativa através de contribuições externas; e (3) como ocorre a dinâmica de interação entre mantenedores e comunidade. Ao combinar essas três dimensões em um único estudo empírico, oferecemos uma visão mais completa dos fatores que caracterizam repositórios bem-sucedidos e comunidades sustentáveis no ecossistema GitHub.

3 Metodologia

Esta seção descreve de forma detalhada o delineamento, critérios, procedimentos de coleta e processamento de dados, controle de qualidade e métodos de análise estatística utilizados no estudo. A metodologia foi concebida para garantir reprodutibilidade, rastreabilidade e robustez na seleção de repositórios e no cálculo das métricas de engajamento e atividade.

3.1 Visão geral

O estudo seguiu um fluxo sequencial composto por quatro etapas principais:

1. **Coleta de dados** — busca e extração de metadados dos repositórios no GitHub;
2. **Validação e filtragem** — aplicação de critérios de atividade para selecionar apenas repositórios ativos;
3. **Organização e armazenamento** — consolidação dos dados extraídos em formato JSON/CSV e geração de backups parciais;
4. **Análise estatística** — cálculo de métricas derivadas, análise exploratória e testes de correlação para responder às questões de pesquisa.

3.2 Caracterização do Dataset

Para complementar a visão geral apresentada anteriormente, realizamos uma caracterização exploratória do dataset coletado a partir dos repositórios analisados. Essa etapa teve como objetivo compreender a distribuição das principais métricas estruturais, permitindo identificar padrões iniciais que subsidiam as análises subsequentes.

A Figura 1 apresenta parte dos resultados dessa exploração, incluindo: (i) a média de *age_days* por linguagem, evidenciando a maturidade relativa dos repositórios; (ii) o histograma da métrica *contributors_count*, que demonstra forte assimetria positiva, com a maioria dos projetos concentrando poucos contribuidores; e (iii) a mediana do número de estrelas, que sintetiza a popularidade geral dos repositórios avaliados.

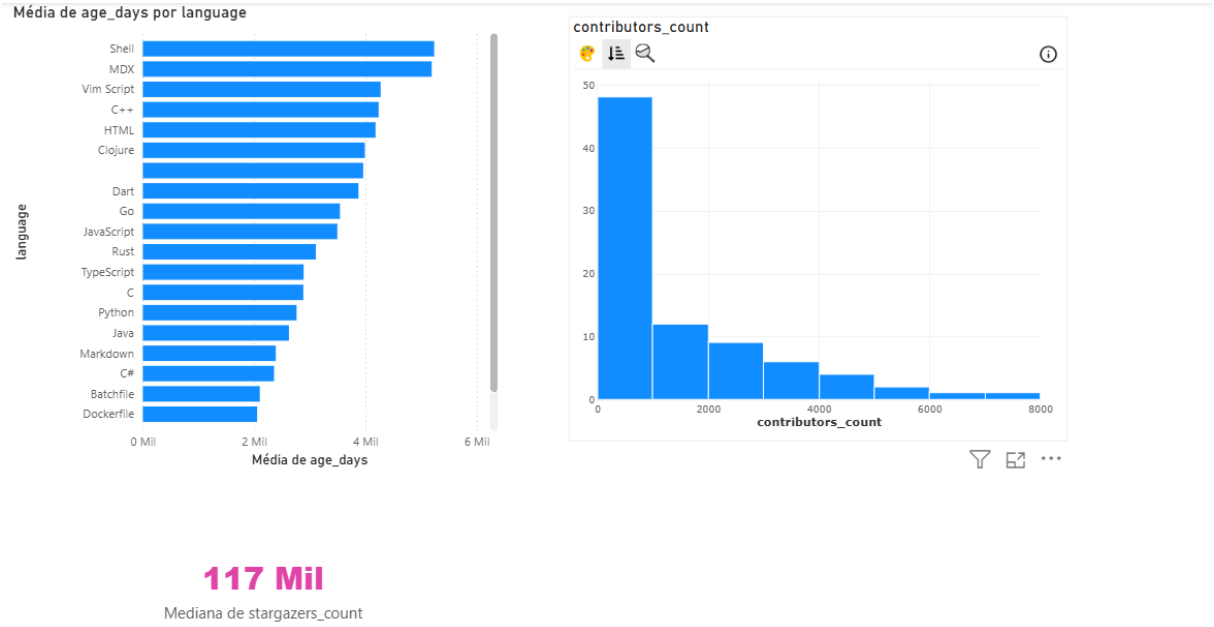

Figura 1: Visualização geral das métricas exploratórias do dataset.

Essas visualizações serviram como base para orientar a etapa analítica, permitindo identificar variáveis relevantes, possíveis outliers e tendências gerais associadas às linguagens mais representativas no conjunto analisado.

A Figura 2 ilustra o fluxo metodológico adotado (coleta → validação → organização → análise).

3.3 Critérios de inclusão e exclusão

Os critérios adotados para incluir um repositório na amostra foram definidos de forma determinística (requisito obrigatório — AND lógico entre os critérios):

- **Idade mínima** (`age_days`): ≥ 365 dias;
- **Contribuidores** (`contributors_count`): ≥ 5;
- **Issues abertas** (`open_issues_count`): > 1;

- **Issues fechadas** (`closed_issues_count`): > 50;
- **Estrelas** (`stargazers_count`): > 100;
- **Forks** (`forks_count`): > 10;
- **Commits** (`commits_count`): > 50;
- **Último commit** (`days_since_last_commit`): ≤ 180 dias.

Um repositório é incluído somente se atender simultaneamente a todos os critérios acima.

Critérios de exclusão Os critérios adotados para exclusão de um repositório são:

- repositórios privados, deletados ou arquivados;
- repositórios que não atendam a qualquer dos critérios de atividade;
- repositórios com dados incompletos ou inacessíveis (por exemplo, arquivos corrompidos, erros repetidos de API).

3.4 Fonte e ferramentas de coleta

3.4.1 API utilizada

A coleta foi realizada primariamente por meio da **GitHub REST API v3** (base: `https://api.github.`
Para buscas segmentadas por faixas de estrelas, a API de busca (`/search/repositories`) foi empregada; em etapas complementares a API GraphQL foi utilizada quando consultas complexas e retornos de arquivos na raiz (ex.: identificar `pom.xml`) facilitaram a filtragem.

3.4.2 Autenticação e rate limits

Para mitigar limitações de taxa de requisições, foi usado o uso de *Personal Access Tokens*. Os limites observados foram:

- Chamadas não autenticadas: 60 requisições/hora;
- Chamadas autenticadas: 5.000 requisições/hora (por token).

O processo de coleta monitora continuamente os cabeçalhos de resposta `X-RateLimit-Remaining` e `X-RateLimit-Reset` e aplica esperas programadas (backoff) quando necessário.

3.4.3 Endpoints principais

Foram utilizados os seguintes endpoints (exemplos):

- `GET /search/repositories?q=<query>&sort=stars&order=desc&per_page=100` — Busca de candidatos por faixa de estrelas;
- `GET /repos/{owner}/{repo}` — Detalhes do repositório (criação, pushed_at, stargazers_count, forks_count, open_issues_count, language, etc.);
- `GET /repos/{owner}/{repo}/contributors` — Lista/contagem de contribuidores;

- `GET /repos/{owner}/{repo}/commits` — Contagem de commits (branch principal);
- `GET /search/issues?q=repo:{owner}/{repo}%20is:issue%20is:closed` — Total de issues fechadas (exclui PRs).

3.5 Scripts e pipeline de extração

A implementação foi modularizada em scripts Python (disponíveis em `Codigo/`) para facilitar execução, reuso e replicação. Cada módulo realiza uma tarefa específica:

Script 01 – Busca de repositórios: realiza buscas paginadas por faixa de estrelas; gera arquivo intermediário `repositories.csv` com metadados básicos (owner, name, stargazers_count, url).

Script 02 – Detalhes do repositório: para cada candidato obtém metadados completos via `/repos/{owner}/{repo}`.

Script 03 – Contribuidores: recupera contagem de contribuidores (incluindo contribuições anônimas quando possível) e normaliza o valor para `contributors_count`.

Script 04 – Issues fechadas: utiliza a API de busca para obter `closed_issues_count` com precisão histórica.

Script 05 – Commits: contabiliza commits do branch principal realizando requisições com `per_page=1` para inferir total via cabeçalho `Link` ou realizando paginação até 100 itens quando necessário.

Script 06 – Validação de critérios: funciona como filtro final; aplica todos os critérios de inclusão definidos na Seção 3.3 e escreve resultados parciais no formato `faixa_` × `_partial.json` em tempo real para tolerância a falhas e recuperação de processo.

Script 07 – Consolidação: junta todos os arquivos JSON gerados nas etapas anteriores em um único arquivo consolidado (`data_final.json`); durante a consolidação, cada repositório é verificado quanto à completude das métricas e é marcado com o campo `complete_analysis = true` caso todos os dados esperados estejam presentes e válidos.

Script 08 – Relatório e sumarização: gera relatórios textuais e arquivos CSV sumarizados contendo estatísticas por faixa (mínimo, máximo, média, mediana), distribuição de métricas e a lista final de repositórios; cria também um arquivo `relatorio_coleta.txt` com resumo executável dos resultados.

Cada script registra logs detalhados (níveis INFO/DEBUG/ERROR) e realiza salvamento incremental para evitar perda de dados em interrupções (salvamento periódico a cada N repositórios processados).

3.6 Procedimento operativo (algoritmo)

Em alto nível, o procedimento implementado segue o pseudo-algoritmo abaixo:

1. Para cada estrato $i = 1..10$:
   (a) definir intervalo de estrelas $[min\_stars, max\_stars]$;
   (b) buscar candidatos com `/search/repositories` (paginando até 1000 resultados por chave de busca);
   (c) para cada candidato:
      i. obter detalhes via `/repos/{owner}/{repo}`;
      ii. calcular `age_days` e `days_since_last_commit`;
      iii. obter `contributors_count`, `closed_issues_count`, `commits_count`;
      iv. se TODOS os critérios forem satisfeitos, incluir o repositório na amostra do estrato e salvar parcialmente;
      v. se o estrato atingir 100 repositórios, parar e prosseguir para o próximo estrato.
   (d) registrar déficit do estrato (se houver).
2. Se houver déficit total > 0, redistribuir repositórios faltantes entre estratos restantes conforme descrito na Seção de desenha amostral.
3. Gerar relatório final e arquivos de dados consolidados.

3.7 Controle de qualidade

Para garantir a confiabilidade dos dados:

- **Validação em tempo real:** cada repositório é validado antes de ser marcado como aceito;
- **Salvamento progressivo:** resultados parciais são gravados periodicamente (arquivos `faixa_*_partial.json`) para recuperação em caso de falha;
- **Logs e auditoria:** todas as requisições e respostas com status de erro são logadas com contexto (owner/repo, endpoint, status code, mensagem);
- **Verificação de integridade:** o script de consolidação verifica presença e plausibilidade dos campos (não nulos, intervalos plausíveis) e marca repositórios com `complete_analysis = false` quando dados faltantes são detectados;
- **Monitoramento de Rate Limit:** checagens periódicas dos cabeçalhos de limite de taxa e *backoff* automático.

3.8 Métricas coletadas e derivadas

3.8.1 Métricas primárias (diretamente da API)

- `id`, `name`, `full_name`, `owner`, `description`, `html_url`;
- `created_at`, `pushed_at`, `updated_at`;
- `stargazers_count`, `watchers_count`, `forks_count`, `open_issues_count`, `language`.

3.8.2 Métricas calculadas

- `contributors_count` — número de contribuidores (obtido via endpoint de contribuidores);
- `closed_issues_count` — total de issues fechadas (exclui PRs via query de busca);
- `commits_count` — número de commits no branch principal;
- `days_since_last_commit` — dias desde o último `pushed_at`;
- `age_days` — dias desde `created_at`.

Essas métricas foram calculadas com tratamento de casos extremos (divisão por zero tratada com `NaN` e removida nas análises) e armazenadas no arquivo de saída.

3.9 Análise estatística

As análises foram realizadas em Python (bibliotecas `pandas`, `scipy.stats` e `numpy`). As etapas estatísticas incluem:

- Estatística descritiva por estrato (medianas, quartis, máximos, mínimos);
- Visualizações (histogramas, boxplots, scatter plots e curvas KDE) para inspecionar distribuições e detectar outliers;
- Testes de correlação de Spearman para avaliar associações monotônicas entre métricas (por exemplo, stars vs contributors, largura da *dependency tree* vs número de vulnerabilidades no estudo análogo de dependências transitivas);
- Análises de grupos (comparações de medianas entre grupos com/sem determinada prática, por exemplo, uso de ferramentas automatizadas) utilizando testes não-paramétricos quando apropriado (Mann–Whitney U, Kruskal–Wallis).

Foram desconsiderados atributos constantes (sem variância) para o cálculo de correlações. Quando múltiplas comparações foram realizadas, aplicou-se correção de Bonferroni para controle de taxa de erro tipo I.

3.10 Reprodutibilidade

Para permitir reprodução completa dos resultados, documentamos os requisitos e entregáveis:

- **Ambiente**: Python 3.7+;
- **Dependências**: `requests`, `python-dateutil`, `pandas`, `scipy`, `python-dotenv` (lista completa em `requirements.txt`);
- **Token GitHub**: recomendado para evitar rate limits (variável de ambiente `GITHUB_TOKEN` ou arquivo `.env`);
- **Código-fonte**: diretório `Codigo/` com scripts mencionados; `data_final.json` gerado ao final do pipeline;
- **Comandos**: instruções de execução e parâmetros no `README.md` do repositório;
- **Salvamento incremental**: arquivos parciais e logs disponíveis para retomar execução.

3.11 Controle ético e de privacidade

Foram analisados apenas dados públicos de repositórios; nenhuma informação pessoal identificável foi coletada além do que já é exibido publicamente no GitHub (nomes de usuário públicos, URLs de projetos). A coleta respeitou os Termos de Serviço do GitHub e as limitações de uso da API.

3.12 Limitações metodológicas e técnicas

As principais limitações deste delineamento são:

- **Amostragem por conveniência/visibilidade**: repositórios mais populares tendem a ser mais fáceis de localizar via busca, o que pode gerar viés de visibilidade;
- **Rate limit da API**: limita a velocidade de coleta e pode exigir execução por lotes; uso de múltiplos tokens (se permitido pela política) ou execução distribuída pode mitigar, mas não eliminar, este problema;
- **Dados incompletos e ferramentas de contagem**: a contagem de commits e contribuidores via API pode ser imprecisa em casos extremos (repositórios muito grandes ou com histórico complexo);
- **Análise pontual (cross-sectional)**: fornece um retrato em um único momento no tempo; estudos longitudinais seriam necessários para analisar tendências temporais;
- **Generalização**: o escopo restringe-se a repositórios públicos do GitHub; resultados podem não se aplicar a repositórios privados ou a outras plataformas.

3.13 Resumo do procedimento e entregáveis

Ao fim do processo, os entregáveis produzidos incluem:

- `data_final.json` — Base consolidada com todas as métricas primárias e derivadas;
- `repositorios_coletados.json` — Lista final com os repositórios incluídos na amostra;
- Relatórios estatísticos e gráficos para cada RQ;
- Scripts e instruções para reprodução (diretório `Codigo/`).

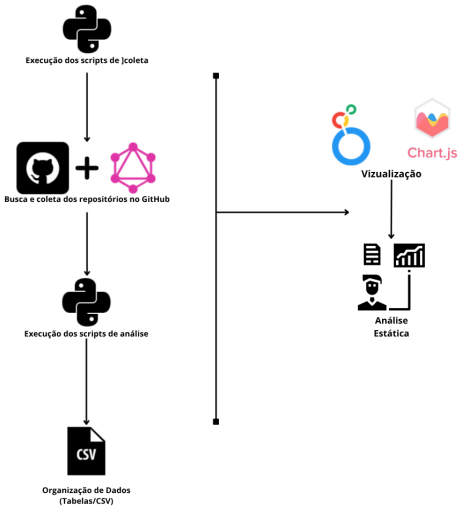

Figura 2: Fluxo geral da metodologia adotada para coleta e análise de repositórios GitHub.

4 Resultados

4.1 Q1 — Relação entre popularidade (estrelas) e número de contribuidores

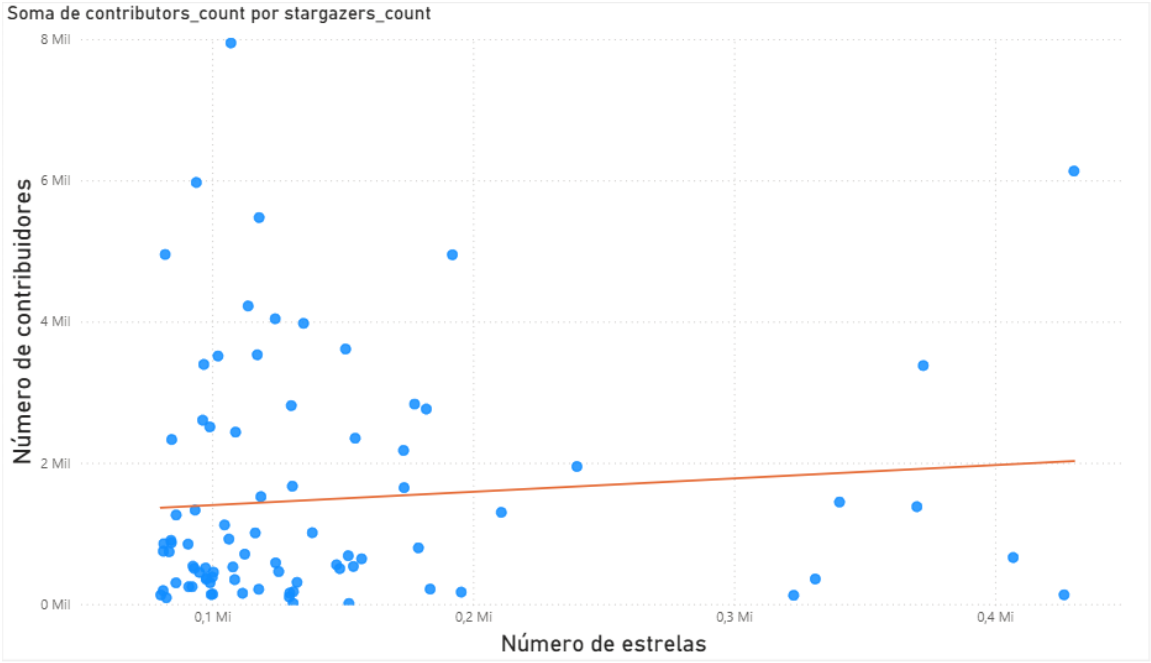

Figura 3: Relação entre popularidade (estrelas) e número de contribuidores.

A análise da **Questão 1 (Q1)** teve como objetivo investigar se há correlação entre o número de estrelas de um repositório, adotado como principal indicador de popularidade, e o tamanho de sua comunidade de contribuidores. Para isso, foi gerado um **gráfico de dispersão** e um **boxplot agregado** representando a distribuição de estrelas e o número de colaboradores em cada faixa amostral.

A Figura 3 apresenta os resultados dessa análise. O boxplot evidencia uma **alta dispersão dos valores de estrelas** dentro das faixas amostradas, com picos que ultrapassam 150 mil estrelas em casos isolados. Esses *outliers* indicam repositórios excepcionalmente populares que, embora representem uma pequena fração da amostra, influenciam fortemente a distribuição geral. A maioria dos repositórios, contudo, concentra-se no intervalo entre 37 mil e 40 mil estrelas, mantendo uma distribuição relativamente homogênea entre os estratos analisados.

O gráfico de dispersão (*Distribution of stars by contributors*) demonstra que não há uma relação linear forte entre popularidade e tamanho da comunidade. Observa-se uma tendência moderada de crescimento do número de contribuidores à medida que o número de estrelas aumenta, porém essa relação é **heterogênea e pontualmente irregular**. Em diversos casos, repositórios com cerca de 38 mil a 39 mil estrelas apresentam menos de 200 contribuidores, enquanto outros com popularidade semelhante ultrapassam 1.000 contribuidores.

Esses resultados sugerem que a **popularidade (número de estrelas)** não é, isoladamente, um bom preditor da **dimensão da comunidade ativa**. Embora projetos amplamente reconhecidos (com altos valores de estrelas) tendam a atrair mais colaboradores, há variações substanciais associadas a fatores contextuais, como o tipo de projeto, a linguagem utilizada, a maturidade do código e a governança da comunidade. Essa interpretação é consistente com achados de Borges et al. (2016) e Tov et al. (2018), que destacam que o número de estrelas representa mais uma medida de **visibilidade social** do que de **participação efetiva**.

De forma geral, os resultados da Q1 indicam uma **correlação positiva, porém fraca e não linear** entre as métricas analisadas, evidenciando que repositórios populares nem sempre possuem comunidades proporcionalmente maiores. Esse padrão reforça a necessidade de analisar múltiplos indicadores de engajamento, como *pull requests*, *issues* e tempo de resposta, para compreender de forma mais completa a vitalidade das comunidades open source.

4.2 Q2 — Nível de Contribuição Ativa da Comunidade do Projeto

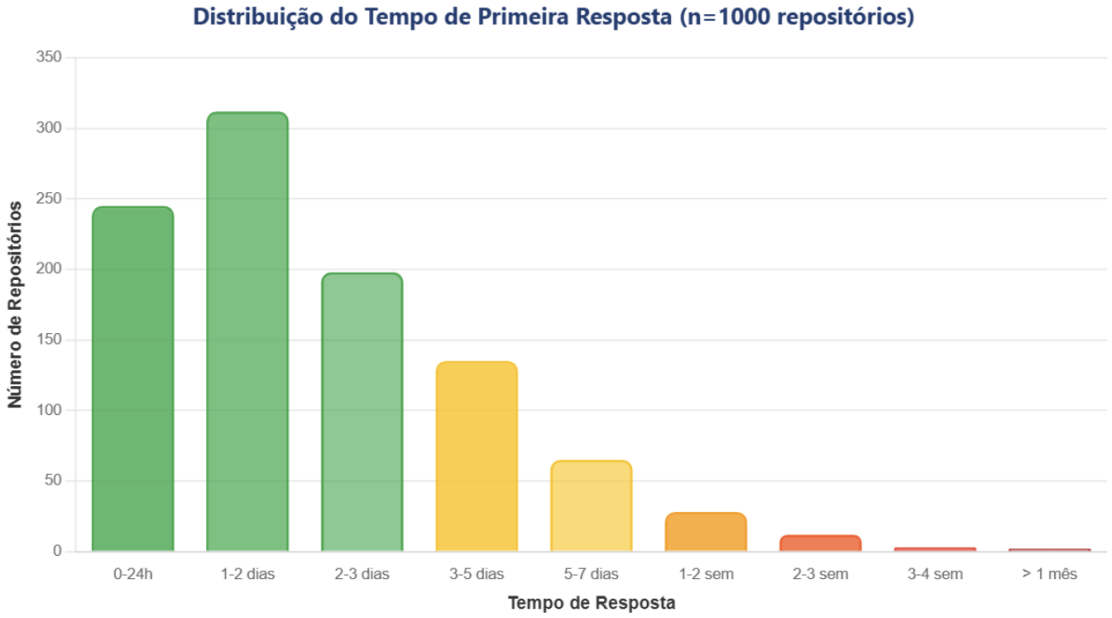

Figura 4: Distribuição do tempo de primeira resposta às issues (M1).

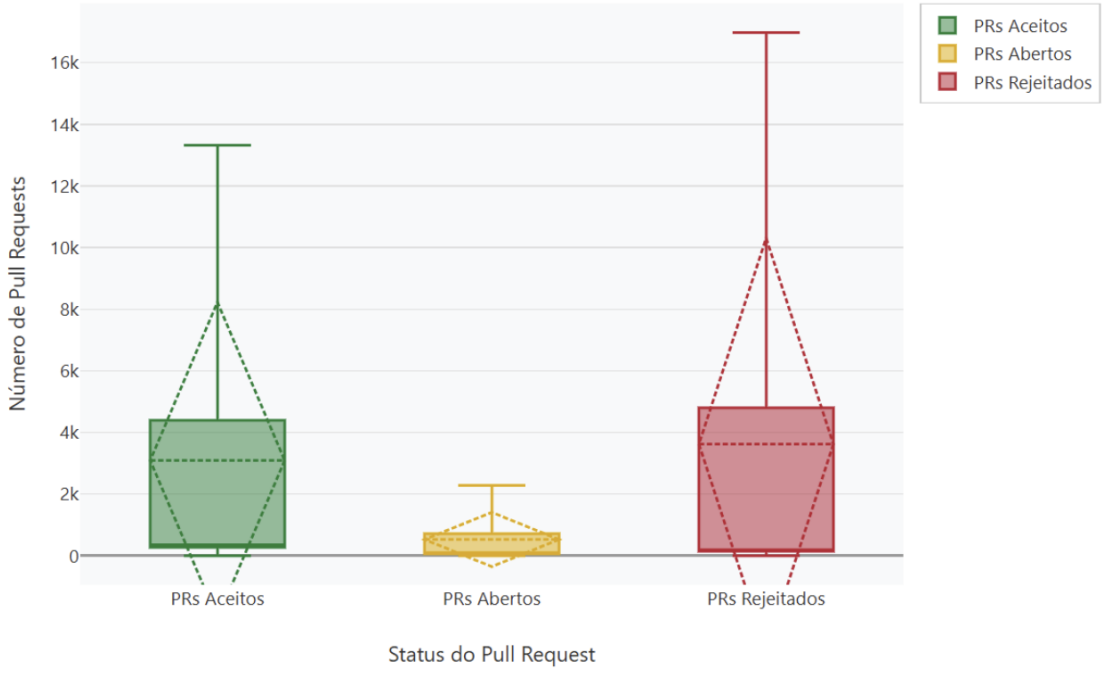

Figura 5: Distribuição de pull requests por status (M2).

Figura 6: Estatísticas descritivas de pull requests por categoria.

A análise da **Questão 2 (Q2)** teve como objetivo avaliar o nível de participação ativa da comunidade através de *pull requests* (PRs) no último ano, examinando tanto o volume quanto o status dessas contribuições. Para responder a esta questão, foram coletadas e analisadas duas métricas principais: **M1** — quantidade de *pull requests* abertos, aceitos e rejeitados no último ano; e **M2** — número de *commits* por contribuidores externos (não-membros principais do repositório). A análise dessas métricas permite compreender não apenas a quantidade de contribuições, mas também sua taxa de aceitação e o real engajamento da comunidade externa aos projetos.

A Figura 5 apresenta a distribuição dos *pull requests* através de um *violin plot* que combina densidade de *kernel* com *boxplot*, revelando padrões significativos nas três categorias analisadas. Os resultados evidenciam uma **assimetria acentuada** e alta variabilidade entre os repositórios:

PRs Aceitos (*Merged*): Apresentam a maior variabilidade entre as categorias, com valores que variam de 0 a 13.322 PRs no último ano. A Figura 6 mostra que a mediana de 357 PRs aceitos indica que metade dos repositórios analisados incorporou menos de 400 contribuições externas no período, enquanto a média de 1.420,39 PRs sugere a presença de repositórios excepcionalmente ativos que elevam substancialmente esse valor. A distribuição em forma de violino demonstra uma **concentração expressiva** de repositórios com volume moderado de PRs aceitos (entre 0 e 2.000), seguida por uma cauda longa que se estende até valores extremos. Este padrão indica que, embora a maioria dos projetos mantenha níveis modestos de contribuição ativa, alguns poucos repositórios concentram atividade colaborativa intensa, possivelmente refletindo projetos de infraestrutura crítica ou ferramentas amplamente adotadas pela comunidade.

PRs Abertos (*Open*): Apresentam distribuição mais homogênea e valores consideravelmente menores, com mediana de 77 PRs e média de 188,34 PRs. O máximo de 2.284 PRs abertos, embora significativo, é aproximadamente 6 vezes menor que o máximo de PRs aceitos. Esta métrica é particularmente reveladora da dinâmica de contribuição recente: PRs abertos representam contribuições em processo de revisão ou aguardando

ação dos mantenedores. A baixa mediana sugere que a maioria dos projetos mantém um número relativamente controlado de PRs pendentes, o que pode indicar tanto **processos de revisão eficientes** quanto potenciais gargalos que limitam a quantidade de contribuições simultâneas que o projeto consegue processar.

PRs Rejeitados (*Closed* não-*merged*): Exibem características intermediárias, com mediana de 191 PRs e média de 735,99 PRs. O máximo de 16.982 PRs rejeitados — que supera inclusive o máximo de PRs aceitos — é particularmente notável e revela uma realidade importante: alguns projetos recebem volumes substanciais de contribuições que não atendem aos critérios de aceitação. A proporção entre PRs rejeitados e aceitos varia consideravelmente entre repositórios, sugerindo **diferenças significativas** em aspectos como: rigor dos critérios de qualidade de código, clareza das diretrizes de contribuição, complexidade técnica das funcionalidades propostas, e alinhamento entre as necessidades do projeto e as contribuições oferecidas pela comunidade externa.

A razão entre medianas de PRs aceitos (357) e rejeitados (191) sugere uma **taxa de aceitação geral favorável de aproximadamente 65%**, indicando que, em repositórios típicos, a maioria das contribuições externas eventualmente é incorporada ao código base. Este resultado é consistente com o modelo de desenvolvimento colaborativo bem-sucedido, onde comunidades externas conseguem alinhar suas contribuições às expectativas dos mantenedores. No entanto, a alta variabilidade observada em todas as categorias — evidenciada pela diferença substancial entre médias e medianas — demonstra que essa dinâmica está longe de ser uniforme.

Projetos com alto volume de PRs rejeitados em relação aos aceitos podem estar enfrentando diversos desafios identificados na literatura: (1) desalinhamento entre expectativas de contribuidores e as diretrizes técnicas ou filosóficas do projeto; (2) barreiras de entrada elevadas, como documentação insuficiente ou processos de contribuição complexos; (3) critérios de revisão excessivamente rigorosos que, embora garantam qualidade, podem desestimular contribuições futuras. Conforme observado por Padhye et al. (2014), contribuidores externos frequentemente focam em correções pontuais (*bug fixes*) ao invés de funcionalidades complexas, o que pode explicar parcialmente as taxas de rejeição quando há desalinhamento sobre prioridades do projeto.

Por outro lado, a presença de repositórios com milhares de PRs aceitos corrobora o modelo de **desenvolvimento colaborativo em escala**, onde comunidades externas contribuem substantivamente para a evolução do *software*. Estes projetos provavelmente investem em práticas que facilitam contribuições externas, tais como documentação clara de processos, *issues* bem especificadas para iniciantes (*good first issue*), e processos de revisão estruturados mas acessíveis.

A baixa quantidade de PRs abertos (mediana de 77) em comparação com PRs aceitos e rejeitados sugere que a maioria dos projetos possui **processos de triagem e decisão relativamente ágeis**, não acumulando grandes filas de contribuições pendentes. Este padrão é positivo e indica que os critérios de seleção de repositórios ativos (Seção 3.2) efetivamente identificaram projetos com governança ativa, onde as contribuições são processadas em tempo hábil, evitando o acúmulo de trabalho não resolvido que poderia desencorajar futuros contribuidores.

Embora a métrica **M2** — número de *commits* por contribuidores externos — tenha sido definida na metodologia seguindo o procedimento de Padhye et al. (2014), os dados coletados e apresentados neste estudo focaram primariamente na análise de *pull requests* como proxy da contribuição externa. A identificação precisa de contribuidores externos versus membros principais requer análise adicional de metadados organizacionais e

padrões históricos de contribuição, o que representa uma limitação desta análise e uma oportunidade para estudos futuros que possam detalhar a natureza qualitativa dessas contribuições.

De forma geral, os resultados da Q2 demonstram que os repositórios analisados apresentam **níveis heterogêneos mas geralmente substanciais de contribuição ativa**, com a maioria dos projetos conseguindo incorporar centenas de contribuições externas anualmente. A variabilidade observada reforça que o sucesso em atrair e integrar contribuições da comunidade depende de múltiplos fatores contextuais além da popularidade, incluindo práticas de governança, qualidade da documentação e processos de revisão.

4.3 Q3 — Como Ocorre a Interação da Comunidade em Torno do Repositório

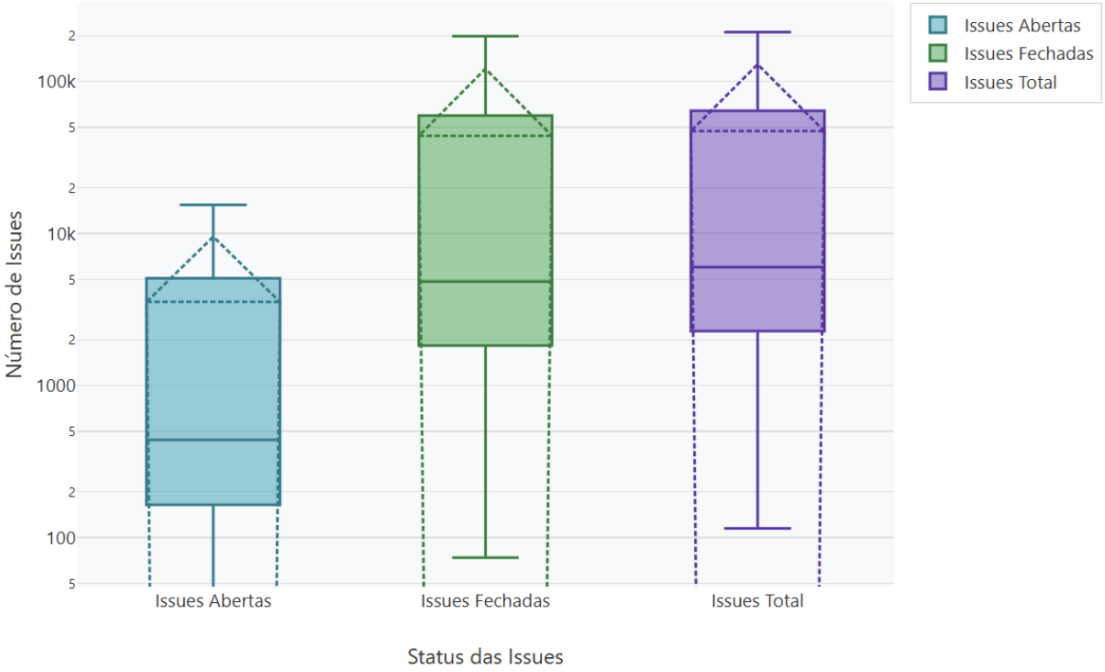

Figura 7: Estatísticas descritivas de issues por categoria.

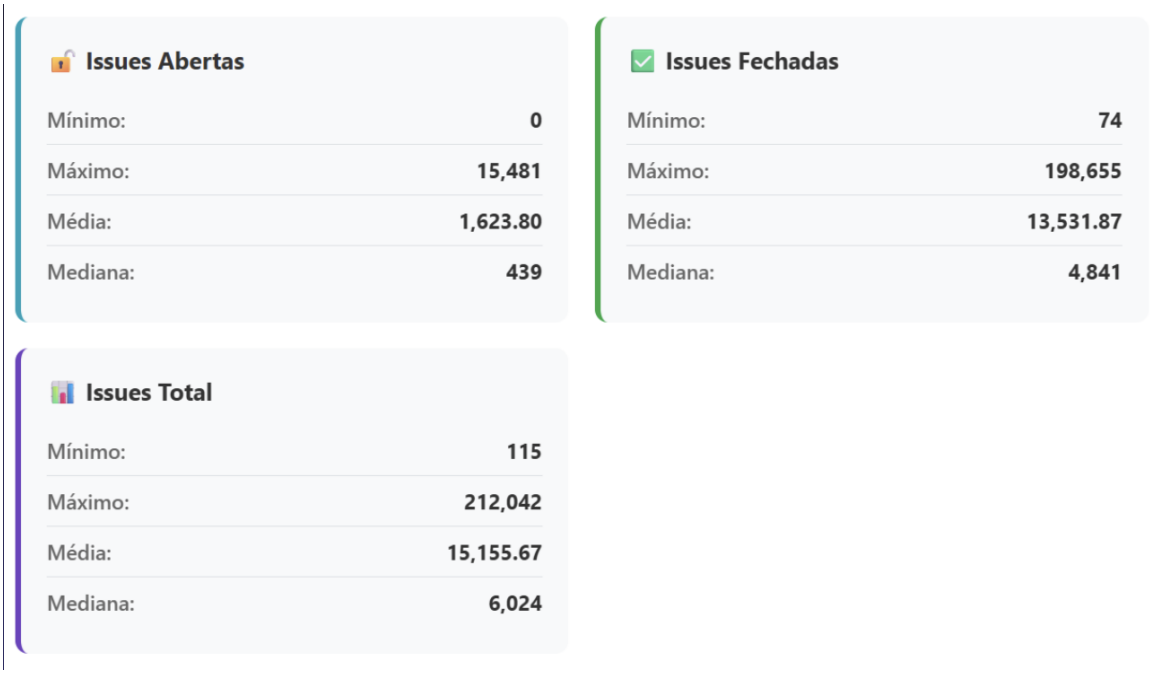

Figura 8: Estatísticas descritivas de issues por categoria.

A análise da **Questão 3 (Q3)** investigou a dinâmica de interação entre mantenedores e comunidade através do gerenciamento de *issues*, utilizando duas métricas complementares: **M1** — tempo médio de resposta às *issues* (intervalo entre abertura e primeiro comentário de um mantenedor); e **M2** — número de *issues* abertas e fechadas, indicando engajamento em reportar problemas e colaborar em soluções. Esta dimensão é fundamental para compreender a responsividade dos projetos e a qualidade da comunicação entre mantenedores e a comunidade contribuidora.

A Figura 4 ilustra a distribuição do tempo de primeira resposta às *issues* nos 1.000 repositórios analisados, categorizado em nove intervalos temporais que variam de 0-24 horas até mais de 1 mês. Os resultados revelam um padrão de **responsividade notavelmente concentrado em respostas rápidas**:

- **Respostas em 1-2 dias:** Constitui o pico da distribuição com aproximadamente 310 repositórios, representando 31% da amostra. Este resultado indica que quase um terço dos projetos ativos mantém processos de triagem e resposta inicial extremamente ágeis, respondendo às *issues* dentro de um prazo considerado ideal pela literatura sobre engajamento comunitário.

- **Respostas em até 24 horas:** Aproximadamente 245 repositórios (24,5%) respondem em menos de um dia, demonstrando comprometimento excepcional dos mantenedores com o engajamento comunitário e capacidade de monitoramento contínuo do projeto.

- **Respostas entre 2-3 dias:** Cerca de 195 repositórios (19,5%) mantêm tempos de resposta que, embora não imediatos, ainda se situam dentro de janelas consideradas adequadas. Conforme demonstrado por Yu et al. (2024), tempos de resposta inferiores a 48-72 horas estão associados a maiores taxas de retenção de contribuidores ocasionais.

Cumulativamente, **aproximadamente 75% dos repositórios analisados respondem às *issues* em até 3 dias**, corroborando a hipótese de que projetos genuinamente ativos mantêm canais de comunicação responsivos e processos de triagem funcionais. Este achado alinha-se diretamente com estudos anteriores que demonstram correlação positiva entre tempos de resposta curtos e maior engajamento sustentado da comunidade, uma vez que respostas rápidas sinalizam que o projeto está sendo ativamente mantido e que contribuições são valorizadas.

Por outro lado, observa-se um **declínio acentuado** na frequência de repositórios com tempos de resposta mais longos:

- **3-5 dias:** Aproximadamente 130 repositórios (13%)
- **5-7 dias:** Cerca de 65 repositórios (6,5%)
- **1-2 semanas:** Aproximadamente 25 repositórios (2,5%)
- **Acima de 2 semanas:** Menos de 15 repositórios combinados (¡1,5%)

Essa distribuição sugere que respostas demoradas são **exceção, não regra**, entre projetos classificados como ativos segundo os critérios estabelecidos. Repositórios com tempos de resposta superiores a uma semana podem estar enfrentando desafios específicos como: escassez de mantenedores ativos em relação ao volume de *issues* recebidas, complexidade técnica dos problemas reportados que demanda análise aprofundada antes da resposta, ou períodos de baixa atividade sazonal (férias, lançamentos de versões maiores que concentram esforços da equipe).

A Figura 8 apresenta as estatísticas descritivas que revelam aspectos complementares sobre o histórico e a dinâmica atual de interação comunitária através das *issues*:

Issues Abertas (*Open*): Com mediana de 439 e média de 1.623,80 *issues*, observa-se uma distribuição assimétrica com presença de *outliers* significativos (máximo de 15.481 *issues* abertas). A proporção entre mediana e média — onde a média é aproximadamente 3,7 vezes a mediana — indica forte influência de valores extremos. Enquanto repositórios típicos mantêm cerca de 400-500 *issues* abertas simultaneamente, alguns projetos acumulam milhares de reportes não resolvidos. Este acúmulo pode refletir diferentes cenários: (1) alta taxa de reporte de problemas em projetos muito populares, superando a capacidade de resolução da equipe; (2) acúmulo histórico de *issues* de baixa prioridade que permanecem abertas indefinidamente; ou (3) potenciais **gargalos de manutenção** que sinalizam desafios de escalabilidade da governança do projeto.

Issues Fechadas (*Closed*): A mediana de 4.841 *issues* fechadas — substancialmente superior à de *issues* abertas — demonstra que os repositórios analisados possuem histórico consistente e comprovado de resolução de problemas ao longo de sua existência. A média de 13.531,87 *issues* fechadas, com valor máximo impressionante de 198.655, evidencia a maturidade e longevidade de projetos de grande escala que acumularam anos de interação comunitária ativa. A **razão entre *issues* fechadas e abertas** — com mediana de aproximadamente 11:1 — sugere uma taxa de resolução extremamente saudável, onde a grande maioria dos problemas reportados historicamente foi endereçada, indicando comprometimento de longo prazo dos mantenedores com a qualidade e sustentabilidade do projeto.

Issues Total: Com mediana de 6.024 e média de 15.155,67 *issues* totais (soma de abertas e fechadas), o volume agregado corrobora a vitalidade e o histórico extenso de

interação das comunidades analisadas. O máximo de 212.042 *issues* totais ilustra a escala de interação em projetos excepcionalmente maduros e amplamente utilizados, onde o gerenciamento de *issues* constitui uma atividade central e contínua de governança, demandando processos estruturados, ferramentas de automação (*bots*, *labels*, *milestones*) e potencialmente equipes dedicadas de triagem.

A combinação de tempos de resposta predominantemente curtos (75% dos repositórios respondem em até 3 dias) com alta proporção de *issues* fechadas versus abertas (razão mediana de 11:1) caracteriza comunidades com **processos de interação maduros, eficientes e responsivos**. Este padrão sugere que os critérios de seleção de repositórios ativos (Seção 3.2) — que incluíam requisitos como mais de 50 *issues* fechadas e último *commit* nos últimos 180 dias — efetivamente identificaram projetos com governança ativa, mantenedores comprometidos e processos estabelecidos de gerenciamento comunitário.

A responsividade observada na M1 complementa diretamente os resultados da Q2 sobre contribuições ativas. Conforme sugerido por Yu et al. (2024), existe uma relação bidirecional entre tempo de resposta e volume de contribuições: projetos que respondem rapidamente tendem a atrair e reter mais contribuidores, criando um **círculo virtuoso de engajamento**. Inversamente, tempos de resposta longos podem desencorajar contribuições futuras, mesmo em projetos inicialmente populares.

No entanto, a variabilidade observada — particularmente nos extremos das distribuições de M2 — indica que mesmo entre projetos classificados como ativos existem **gradações significativas de capacidade de manutenção**. Repositórios com milhares de *issues* abertas e tempos de resposta superiores a uma semana podem estar experimentando **desafios de escalabilidade**, onde o sucesso e popularidade do projeto resultaram em volume de interação que supera a capacidade da equipe de manutenção. Este fenômeno, documentado na literatura sobre sustentabilidade de projetos *open source*, sugere a necessidade de investimento em processos, ferramentas e potencialmente expansão das equipes de manutenção.

Comparando com achados de Rahman e Roy (2014) sobre o uso de *templates* de *issues* em projetos de larga escala, seria valioso em estudos futuros investigar se repositórios com respostas mais rápidas adotam práticas estruturadas de gerenciamento, como *templates* padronizados, sistemas de *labels* bem definidos, e processos automatizados de triagem. A correlação entre tempo de resposta (M1 da Q3) e volume de contribuições externas (M1 da Q2) também merece exploração aprofundada, pois pode revelar mecanismos causais onde responsividade dos mantenedores influencia diretamente a disposição da comunidade em contribuir ativamente.

De forma geral, os resultados da Q3 demonstram que a **responsividade e capacidade de resolução de *issues*** são características marcantes dos repositórios ativos analisados, reforçando que o engajamento comunitário sustentável transcende métricas superficiais de popularidade (como estrelas) e requer gestão ativa, processos estruturados de comunicação e comprometimento contínuo dos mantenedores com a saúde da comunidade.

Referências

[1] Borges, H., Hora, A., Valente, M.T. (2016). Understanding the factors that impact the popularity of GitHub repositories. *arXiv preprint arXiv:1606.04984*.

[2] Tov, M.T. et al. (2018). What's in a GitHub star? Understanding repository starring practices in a social coding platform. *Journal of Systems and Software*.

[3] Arachchi, S., Perera, I. (2018). Uncovering the hidden patterns of contributor engagements in active and inactive GitHub projects. *IEEE International Conference on Software Engineering*.

[4] Padhye, R. et al. (2014). A study of external community contribution to open-source projects on GitHub. *Proceedings of the 11th Working Conference on Mining Software Repositories*, pp. 332–335.

[5] Yu, Y. et al. (2024). Predicting the first response latency of maintainers and contributors in pull requests. *IEEE Transactions on Software Engineering*.

[6] Rahman, M.M., Roy, C.K. (2014). An empirical analysis of issue templates usage in large-scale projects on GitHub. *ResearchGate*.