
**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

**INSTITUTO DE CIÊNCIAS EXATAS E INFORMÁTICA**

**UNIDADE EDUCACIONAL PRAÇA DA LIBERDADE**

**Bacharelado em Engenharia de Software**

**Carla d'Abreu Martins Vieira**

**Daniel Lyncon Gonçalves de Souza**

# CARACTERÍSTICAS DE REPOSITÓRIOS POPULARES

Laboratório de Experimentação de Software

Belo Horizonte

2021

**Carla d'Abreu Martins Vieira**

**Daniel Lyncon Gonçalves de Souza**

# CARACTERÍSTICAS DE REPOSITÓRIOS POPULARES

Laboratório de Experimentação de Software

Trabalho prático de estudo sobre como os sistemas populares open-source são desenvolvidos e suas outras características.

Professor José Laerte Pires Xavier Junior

Belo Horizonte

2021

# SUMÁRIO

## 1. Introdução

Este trabalho tem por objetivo estudar os níveis de atividade e engajamento dos 1000 projetos open source mais populares com base no tempo de existência e tecnologias utilizadas. Dessa forma, pretende-se analisar com quais linguagens eles são desenvolvidos, com que frequência recebem contribuição externa, com qual frequência lançam novas versões, entre outras características.

Foram determinadas 7 questões de pesquisa e, para cada uma delas, foram elaboradas métricas com base nos dados fornecidos pela API do **GitHub** (plataforma de hospedagem de repositórios). São elas:

- **RQ 01 -** Sistemas populares são maduros/antigos?
  - **Métrica:** idade do repositório.
- **RQ 02 -** Sistemas populares recebem muita contribuição externa?
  - **Métrica:** total de pull requests aceitos.
- **RQ 03 -** Sistemas populares lançam releases com frequência?
  - **Métrica:** total de releases.
- **RQ 04 -** Sistemas populares são atualizados com frequência?
  - **Métrica:** tempo desde a última atualização.
- **RQ 05 -** Sistemas populares são escritos nas linguagens mais populares ?
  - **Métrica:** linguagem primária de cada um desses repositórios.
- **RQ 06 -** Sistemas populares possuem um alto percentual de issues fechadas?
  - **Métrica:** razão entre número de issues fechadas pelo total de issues.
- **RQ 07 -** Sistemas escritos em linguagens mais populares recebem mais contribuição externa, lançam mais releases e são atualizados com mais frequência?
  - **Métrica:** pull requests aceitos por linguagem e releases por linguagem.

Com estas questões e métricas espera-se encontrar um padrão entre os repositórios mais populares da plataforma. Considerando que a popularidade dos

repositórios é dada com base no número de estrelas de cada um, imagina-se que outras métricas sejam semelhantes entre eles.

Com base em deduções e inferências foram levantadas algumas hipóteses dos resultados que serão encontrados após a coleta e análise dos dados. São elas:

- Os repositórios mais populares são antigos e utilizam linguagens menos populares atualmente, mas que já foram um dia.
- Os repositórios mais populares utilizam linguagens populares atualmente.
- Os repositórios populares possuem mais pull requests aceitas e são atualizados com frequência.
- Os repositórios antigos possuem mais releases e maior taxa de issues fechadas.

## 2. Metodologia

Para a mineração dos dados, foi construída uma query GraphQL em Python que solicita à API do Github. A query utilizada está disponível em /mining_script.py e representada a seguir. Foi necessária a elaboração de uma estratégia para solicitação de dados em pequenos números para evitar problemas com a limitação das requisições, utilizando então o valor do *endCursor* para criar um sistema de paginação .

```
import dotenv
import os
import requests
import pandas as pd

# Defines number of nodes
num_nodes_total=1000
num_nodes_request=5

# Retrieves Github API Token from .env
dotenv.load_dotenv(dotenv.find_dotenv())
```

```
TOKEN = os.getenv("GITHUB_ACCESS_TOKEN")
HEADERS = {
    'Content-Type': 'application/json',
    'Authorization': f'bearer {TOKEN}'
}
URL = 'https://api.github.com/graphql'

# Query to be made on the current test
def create_query(last_cursor):
    QUERY = """
     query github {
        search (query: "stars:>10000", type:REPOSITORY,
first:"""+str(num_nodes_request)+""") {
            pageInfo {
                endCursor
                }
            nodes {
              ... on Repository {
                nameWithOwner
                createdAt  # RQ 01
                pushedAt
                primaryLanguage {
                  name
                }
                stargazers {
                  totalCount
                }
                releases {
                  totalCount
                } # RQ 03
                mergedPRs: pullRequests(states: MERGED) {
                  totalCount
                }
                closedIssues: issues(states: CLOSED) {
                  totalCount
                }
                totalIssues: issues {
                  totalCount
                }
              }
            }
```

```
        }
    }
    """
    if last_cursor is not None:
        QUERY = """
         query github {
            search (query: "stars:>10000", type:REPOSITORY,
first:"""+str(num_nodes_request)+""",
after:"""+"\""+(last_cursor)+"\""+""") {
                pageInfo {
                    endCursor
                    }
                nodes {
                  ... on Repository {
                    nameWithOwner
                    createdAt  # RQ 01
                    pushedAt
                    primaryLanguage {
                      name
                    }
                    stargazers {
                      totalCount
                    }
                    releases {
                      totalCount
                    } # RQ 03
                    mergedPRs: pullRequests(states: MERGED) {
                      totalCount
                    }
                    closedIssues: issues(states: CLOSED) {
                      totalCount
                    }
                    totalIssues: issues {
                      totalCount
                    }
                  }
                }
            }
        }
        """
    return QUERY
```

```
# RQ 01 -> ON REPOSITORY/CREATED AT
# RQ 02 -> pullRequests(merged)/totalCount
# RQ 03 -> releases/totalCount
# RQ 04 -> ON REPOSITORY/PUSHED AT
# RQ 05 -> primaryLanguage / name
# RQ 06 -> closedIssues / totalCount

last_cursor = None
nodes = pd.DataFrame()
pages = num_nodes_total // num_nodes_request
print(f"It will take {pages} pages")
for page in range(pages):

    condition = True
    while condition:

        try:
            response = requests.post(f'{URL}', json={'query':
create_query(last_cursor)}, headers=HEADERS)
            response.raise_for_status()
            data = dict(response.json())

last_cursor=data['data']['search']['pageInfo']['endCursor']
            df = pd.DataFrame(data['data']['search']['nodes'])
            nodes = nodes.append(df)

        except requests.exceptions.ConnectionError:
            print(f'Connection error during the request')

        except requests.exceptions.HTTPError:
            print(f'HTTP request error. STATUS:
{response.status_code}')

        except FileNotFoundError:
            print(f'File not found.')

        else:
            print(f"Page {page}/{pages} succeeded!")
            condition = False
```

```
nodes.to_csv(os.path.abspath(os.getcwd()) +
'/export_dataframe.csv', index=False, header=True)
print("Successful mining! Saved csv with mining results")
```

Os dados utilizados no estudo são referentes apenas aos repositórios. São eles:

- **nameWithOwner:** nome do repositório e do seu proprietário;
- **createdAt:** data de criação do repositório;
- **pushedAt:** data da última atualização do repositório;
- **primaryLanguage:** nome da linguagem principal utilizada no repositório;
- **stargazers:** quantidade de estrelas do repositório;
- **releases:** quantidade de lançamentos de novas versões do projeto;
- **mergedPRs:** quantidade de pull requests aceitos;
- **closedIssues:** quantidade de problemas resolvidos no repositório;
- **totalIssues:** quantidade total de problemas no repositório.

Utilizando a Query apresentada, foi produzido um arquivo CSV com os primeiros 1000 repositórios com mais estrelas do GitHub, que pode ser encontrado em /export_dataframe.csv . Assim, para a limpeza e manipulação dos dados com intuito de realizar uma análise final dos resultados, este arquivo CSV foi importado para o Google Sheets, onde o resultado final, junto às fórmulas utilizadas podem ser acessados em Acesso ao GoogleSheets.

## 3. Resultados

### 3.1. (RQ 01) Idade do repositório

Para analisar o quesito de idade do repositório, utilizamos do campo **createdAt,** que tem a data da criação do repositório. Com a data de criação,

calculamos a diferença entre a data da criação e a data atual e convertemos esse valor em meses. Com os dados de quantos meses cada repositório tinha de idade, tivemos uma média de 73,892 meses, mediana de 73 e e moda de 80. O desvio padrão encontrado foi de 31,12544183. Temos então que a categoria com mais repositórios foi de 63,2 a 93,8 meses, que representa 35,10% do total de repositórios analisados. Na Figura 1, pode ser observado o gráfico comparativo entre as categorias de idade dos repositórios.

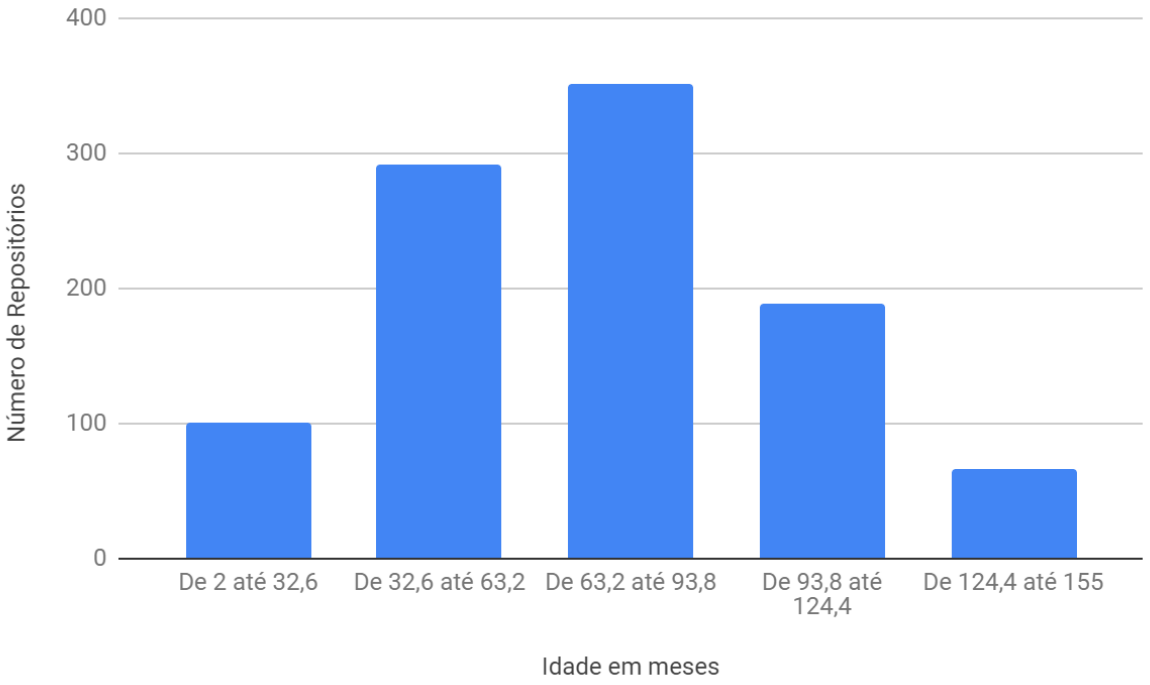


*Figura 1: Gráfico Idade em meses x Número de Repositórios*

### 3.2. (RQ 02) Total de Pull Requests Aceitos

Já para número de *pull requests* aceitos, temos um desvio padrão muito maior, de 5047,481146. Isso acontece por que temos muitos repositórios com o valor de 0 *PRs* aceitos, que é o valor da moda encontrado,, enquanto temos alguns repositórios pontuais com um número muito alto, como o *Homebrew/homebrew-cask* que tem 79517. Com isso, calculamos um valor de média de 1684,54, mediana de 336,5. A grande maior parte dos repositórios apresentam até 15903,4 PRs aceitos, como pode ser observado na Figura 2.

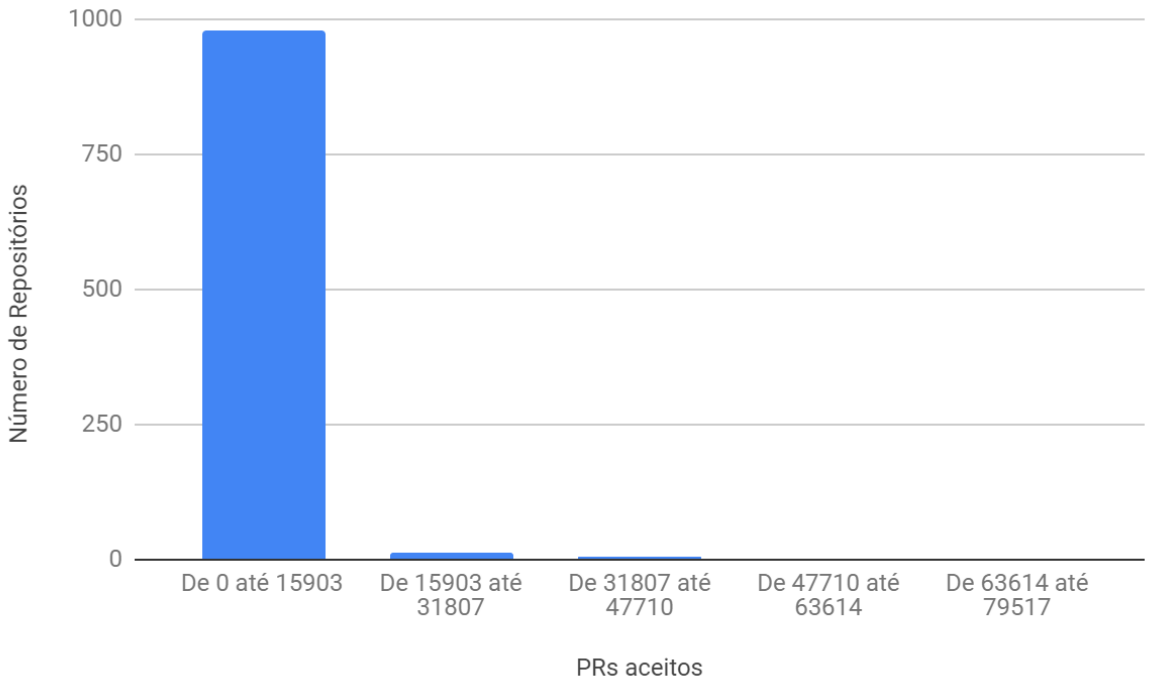


*Figura 2: Gráfico PRs aceitos x Número de Repositórios*

A fim de analisar melhor os repositórios que em sua grande maioria tem uma quantidade não tão expressiva de PRs aceitos, temos que pelo menos 10% dos 1000 repositórios analisados têm menos de 25 PRs aceitos, como pode ser observado na Figura 3.

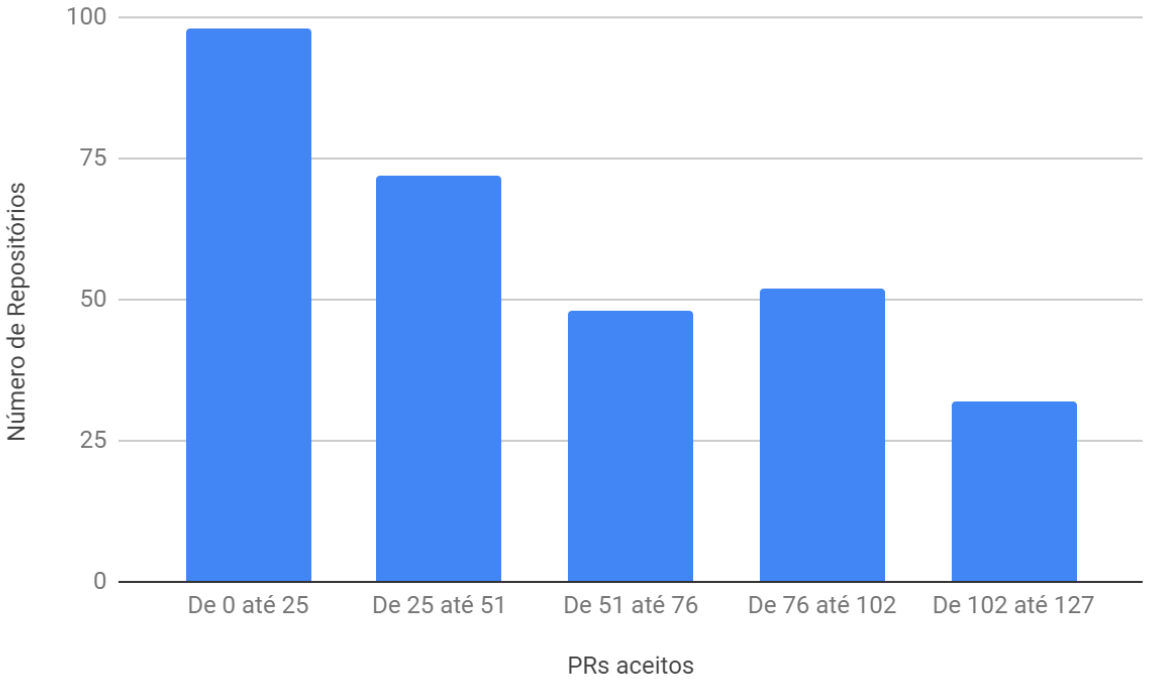


*Figura 3: Gráfico PRs aceitos até 127 x Número de Repositórios*

### 3.3. (RQ 03) Total de Releases

Para número de releases, também temos um valor grande de desvio padrão, mesmo que não tão grande quanto de PRs aceitos, no valor de 93,0438. A moda encontrada também foi 0, e o valor da média foi 47,457 e da mediana 47,457. Assim, a grande maioria dos repositórios apresenta até 189,4 releases.

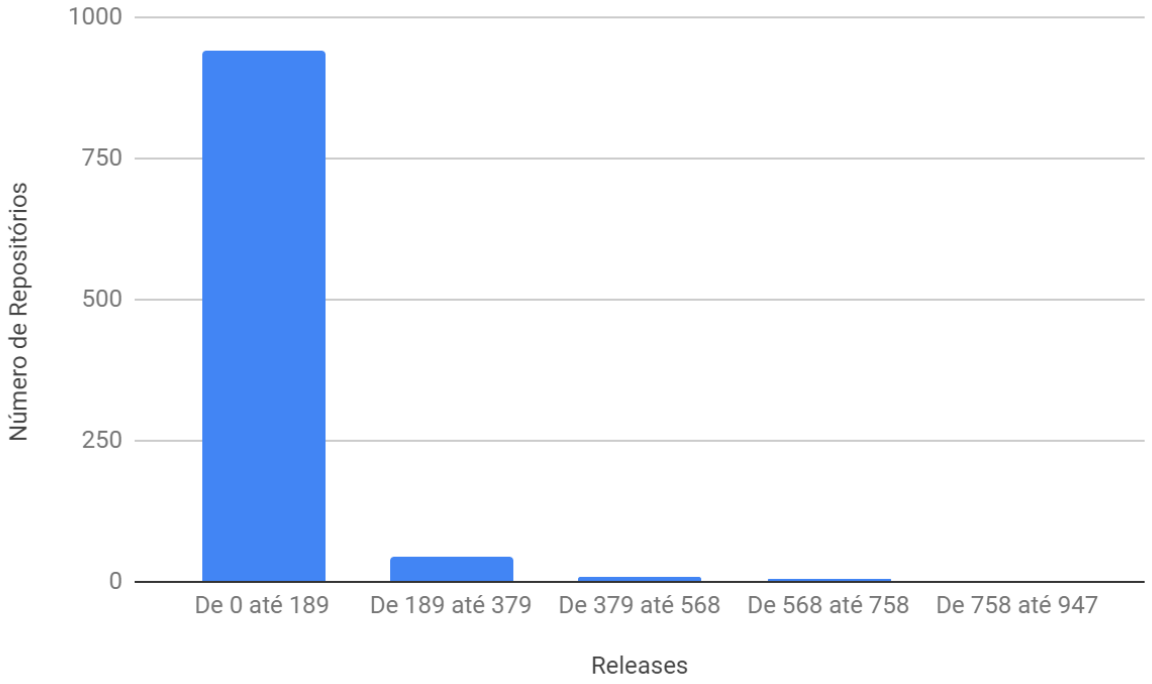


*Figura 4: Gráfico Releases x Número de Repositórios*

Analisando os repositórios com menos número de releases, temos que 46,30% , ou seja, 463 repositórios, têm menos de 8 releases, como pode ser observado na Figura 5.

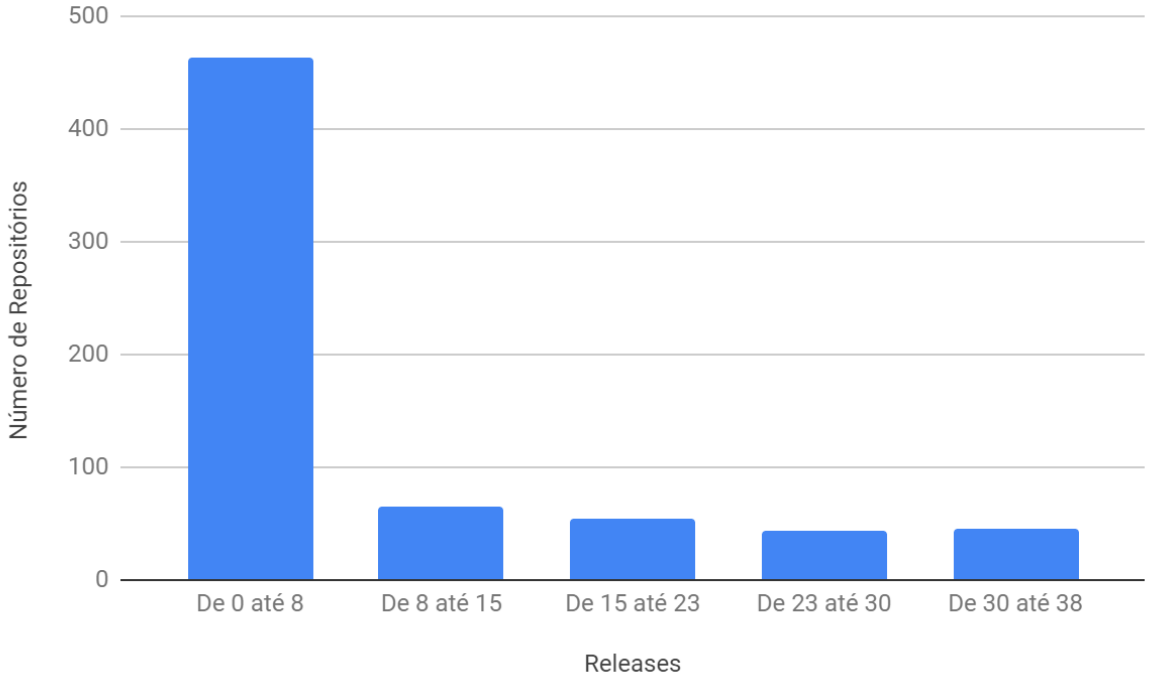


*Figura 5: Gráfico Releases até 38 x Número de Repositórios*

### 3.4. (RQ 04) Tempo desde a última atualização

Para analisar o quesito de tempo desde a última atualização, utilizamos do campo **pushedAt**. Com essa data, calculamos a diferença entre o *pushedAt* e a data da mineração dos dados  e convertemos esse valor em dias. Com esses dados, tivemos uma média de 43,043 dias, mediana de 7 e e moda de 4. O desvio padrão encontrado foi de 126,0380289. Como repositório com mais tempo desde a última atualização, temos o oxford-cs-deepnlp-2017/lectures, com 1363 dias, sendo um extremo discrepante aos demais repositórios onde grande parte é atualizado com maior frequência, como pode ser observado na Figura 6.

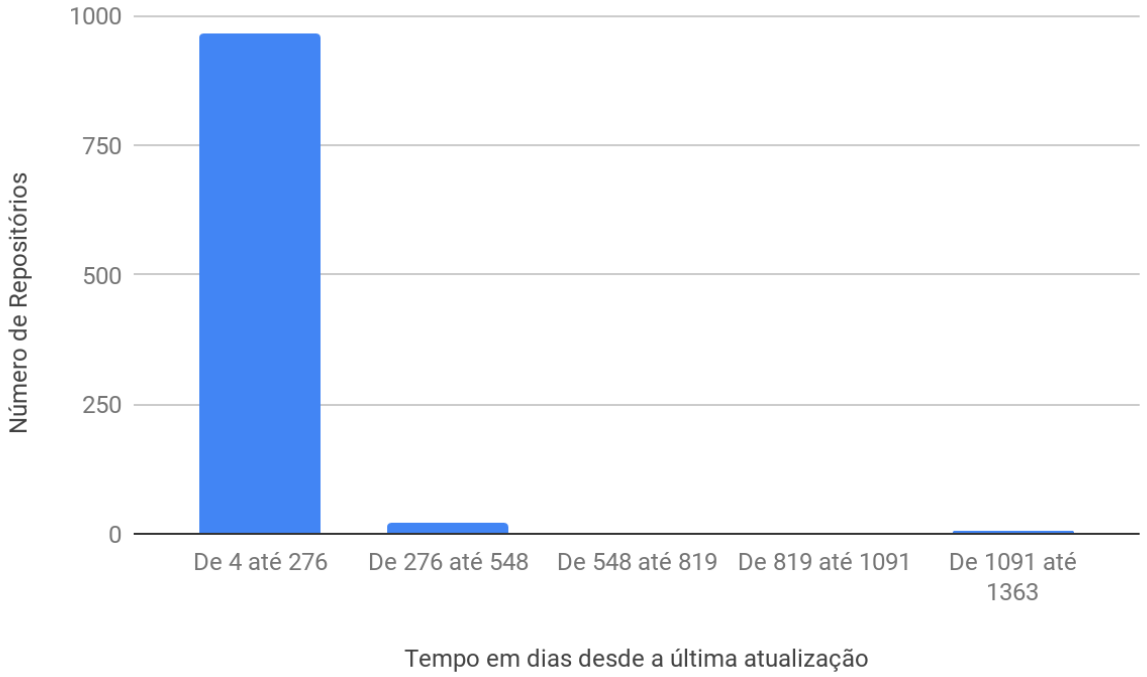


*Figura 6: Gráfico Tempos em dias desde a última atualização  x Número de Repositórios*

Assim, temos que aproximadamente metade dos repositórios analisados são atualizados a menos de uma semana, como pode ser observado na Figura 7

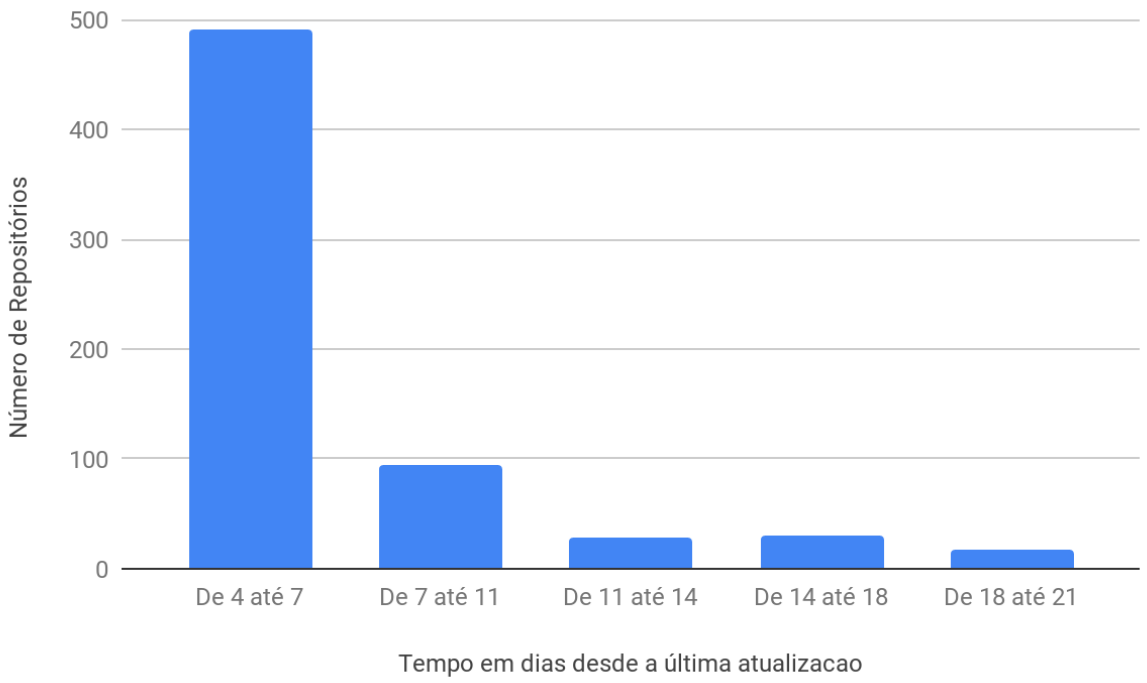


*Figura 7: Gráfico Tempos em dias desde a última atualização até 21 x Número de Repositórios*

### 3.5. (RQ 05) Linguagem primária do repositório

Entre os repositórios os quais é identificado uma linguagem primária, foi possível observar um destaque maior para as linguagens que apareceram na respectiva ordem no ranking: JavaScript (31,3%), Python (11,5%), Java (9,0%), TypeScript (7,5%) e Go (7,5%). Certa de 22,6% dos repositórios com linguagens identificadas apresentam uma das 35 linguagens que tem representação igual ou menor que 1%.

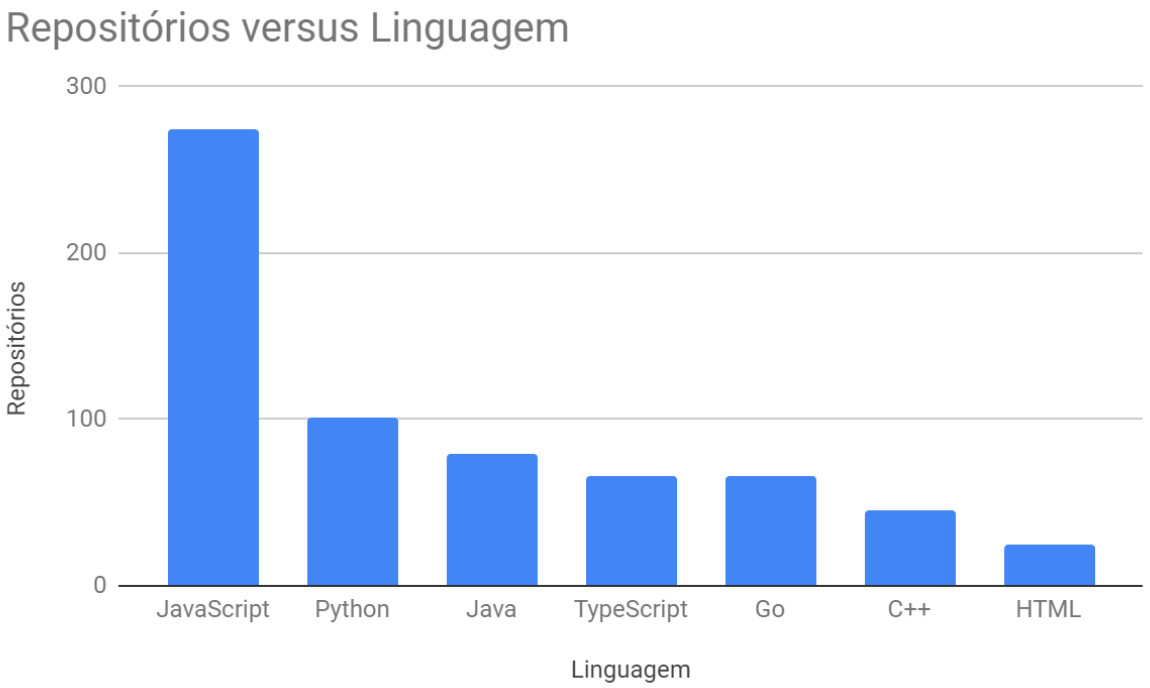


*Figura 8: Gráfico Linguagem primária x Número de Repositórios*

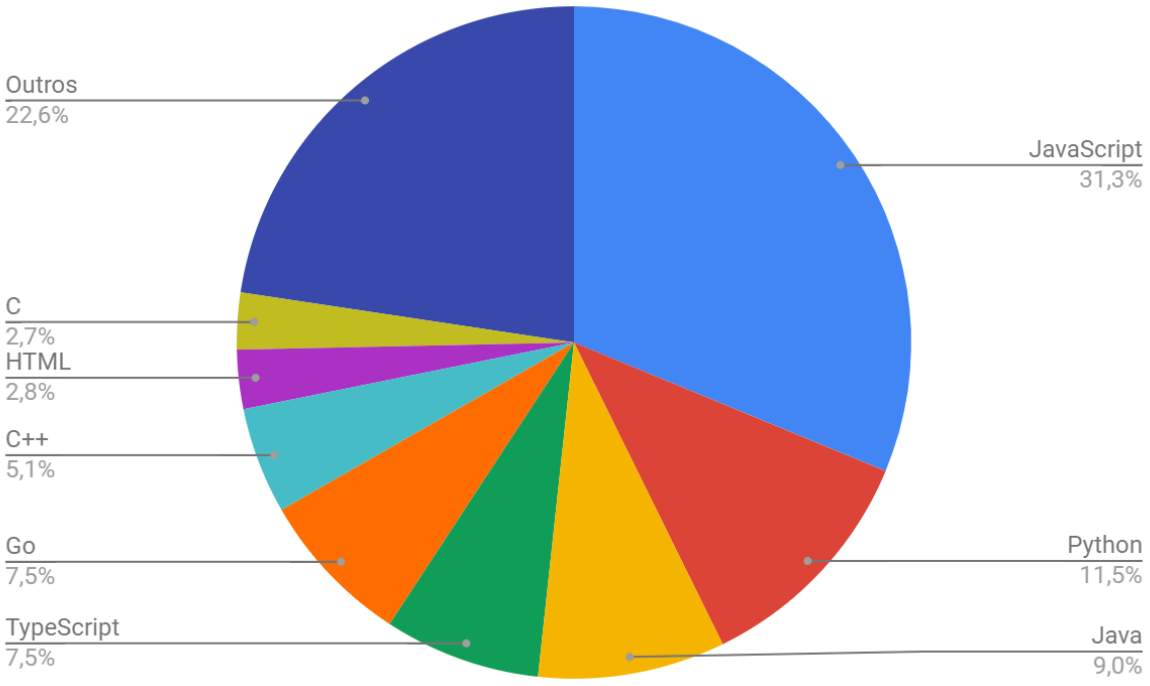


*Figura 9: Gráfico Linguagem primária x Número de Repositórios*

### 3.6. (RQ 06) Razão de issues fechadas pelo total de issues

Utilizando a razão entre a quantidade de issues fechados pelo total de issues, temos a porcentagem de fechamento de issues de cada repositório. Esses dados apresentaram média de 80,97%, mediana de 86,60% e moda 100,00%, com o valor de 18,80%. Com isso, e junto à observação da Figura 10, temos que a maior parte dos repositórios tem um índice maior de 80% de fechamento de issues.

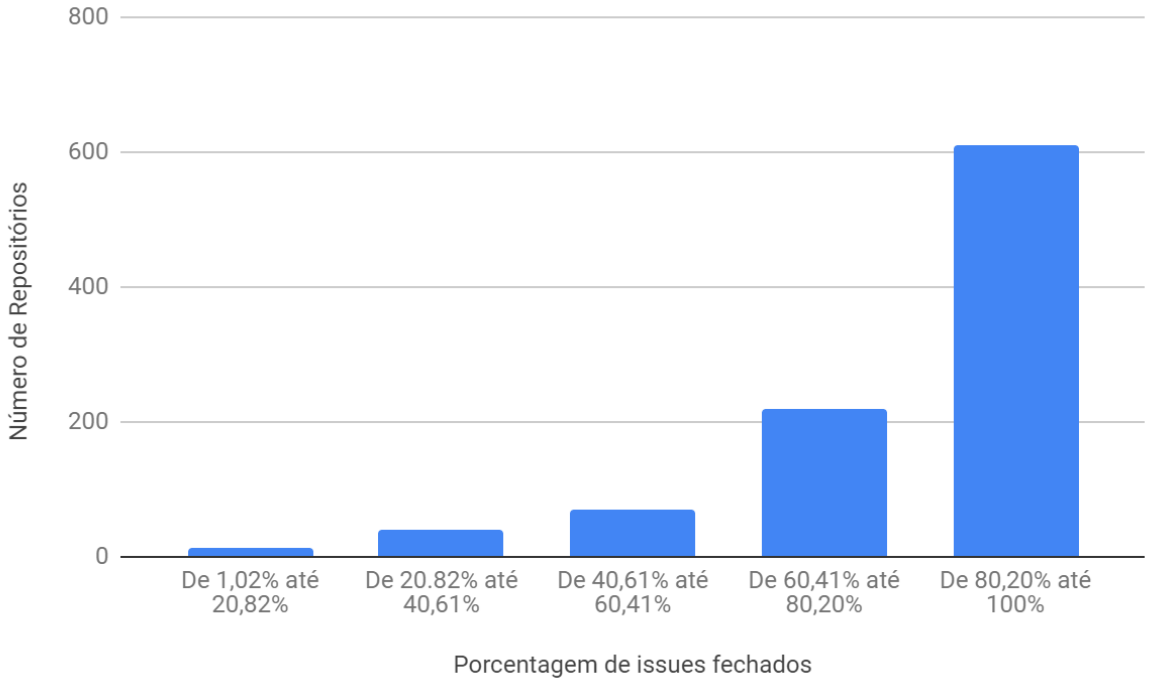


*Figura 10: Gráfico Porcentagem de Issues fechadas x Número de Repositórios*

### 3.7. (RQ 07) Nível de atividade baseado na linguagem primária

Foi feita uma análise comparativa das cinco linguagens mais representativas nos dados coletados, sendo elas JavaScript (31,3%), Python (11,5%), Java (9,0%), TypeScript (7,5%) e Go (7,5%), com as médias encontradas para os repositórios de maneira geral. Para o número de releases, temos que todas as linguagens analisadas apresentam uma média menor que a média geral, com um destaque para Java, com média de 322, como pode ser observado na Figura 11.

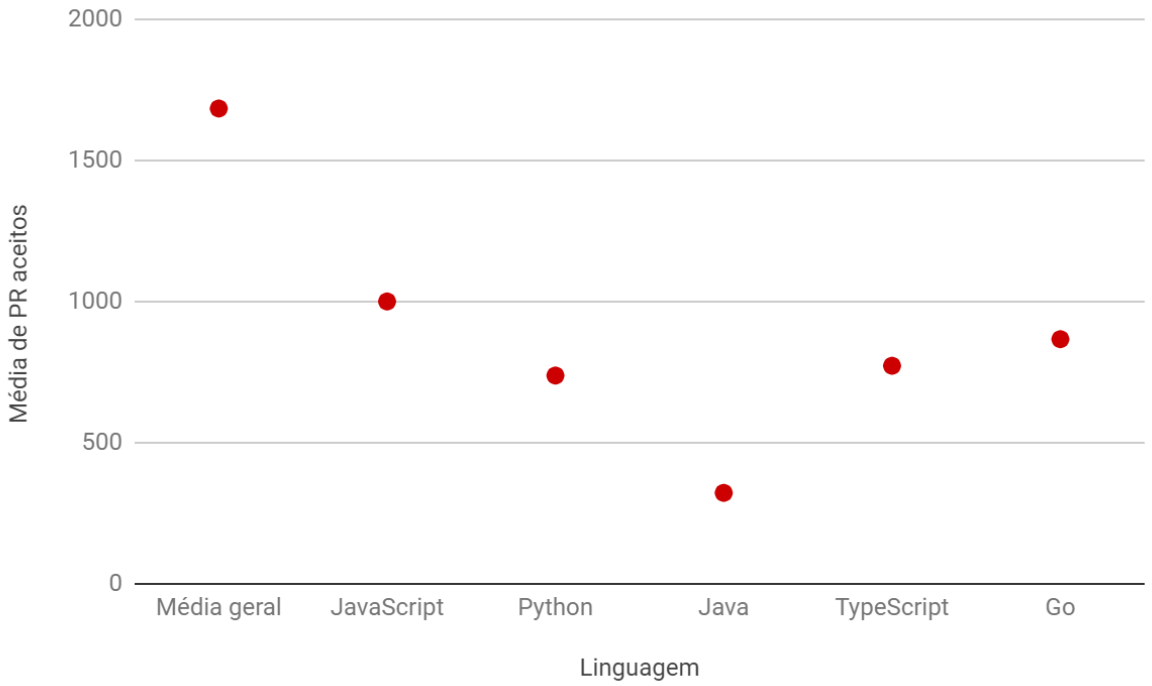


*Figura 11: Linguagem x Média de PRs aceitos*

Para a média de release, temos o destaque para JavaScript, que é a única com média maior que a média geral, com o valor de 60 releases por repositório, no comparativo da Figura 12.

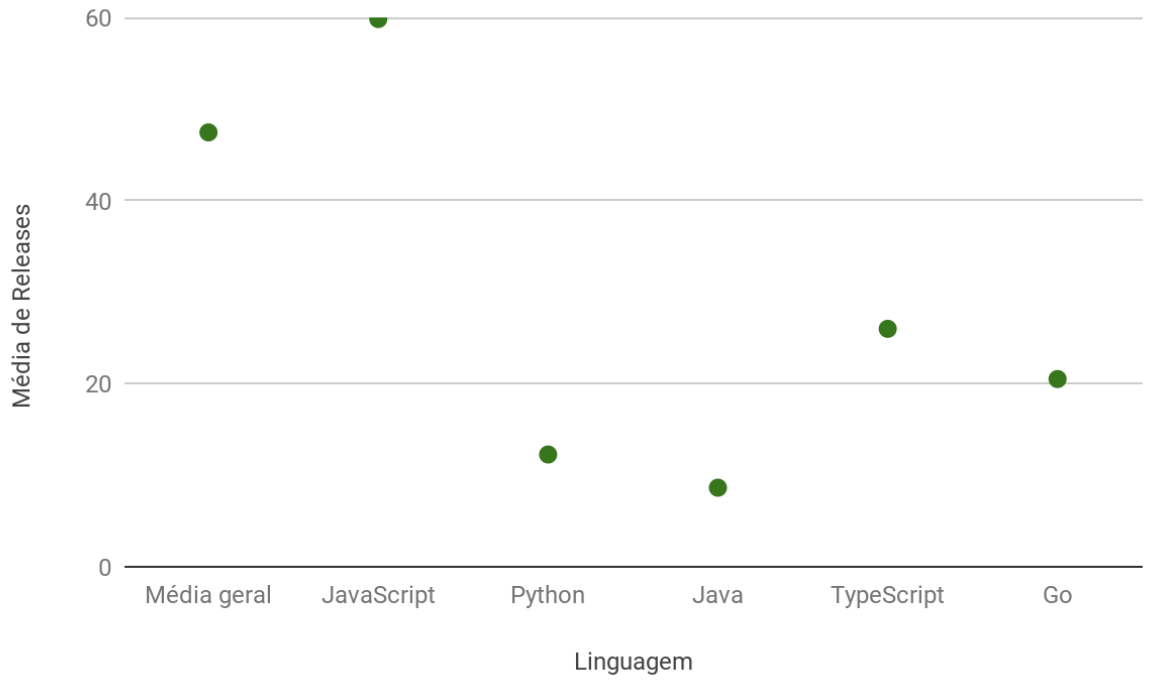


*Figura 12: Linguagem x Média de Releases*

Já para o tempo médio desde a última atualização em dias, tempos que todas as cinco linguagens tem uma média menor que a média geral dos repositórios, na respectiva ordem de representatividade, como pode ser observado na Figura 13.

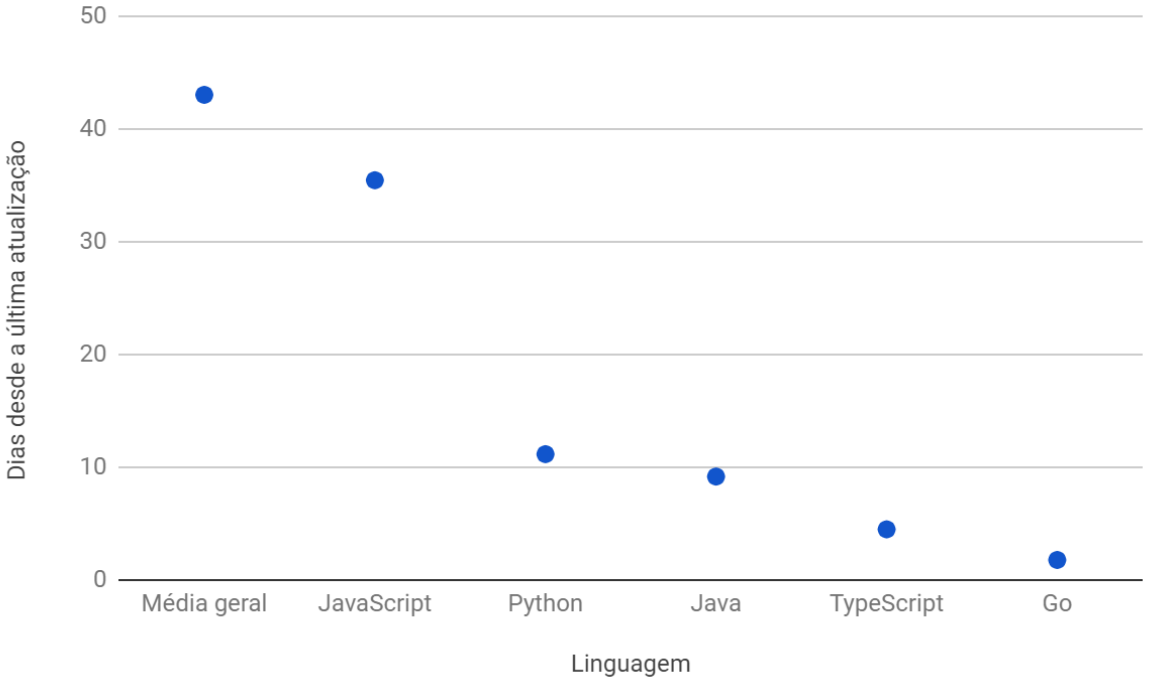

*Figura 13: Gráfico Linguagem x Média de dias desde a última atualização*

## 4. Conclusão

Após a realização dos experimentos e da análise dos resultados é possível chegar a algumas conclusões para responder às questões de pesquisa e validar ou invalidar as hipóteses deste estudo.

Nota-se que a idade dos repositórios não é um fator determinante da sua popularidade, uma vez que há uma maior distribuição entre idade dos repositórios listados. Por sua vez, as linguagens primárias, a quantidade de lançamentos, quantidade de pull requests aceitas e percentual de problemas resolvidos são características marcantes dos repositórios mais populares. Ou seja, estes dados que indicam o nível de atividade e engajamento dos repositórios estão mais ligados a sua popularidade. De tal forma, as linguagens influenciam diretamente neste engajamento. Sendo assim, quase todas as questões de pesquisa se confirmam como fatores importantes para um repositório de sucesso, exceto pelo tempo de existência dos repositórios que se mostrou pouco influente.