Pontifícia Universidade Católica de Minas Gerais
Departamento de Engenharia de Software e Sistemas de Informação
Disciplina: Laboratório de Experimentação de Software
Professor: Laerte Xavier

Características de Sistemas Populares GitHub

João Américo Martins Caiafa de Andrade

joao.andrade@sga.pucminas.br

1. Introdução

GitHub é uma plataforma de hospedagem de código-fonte com controle de versão usando o Git. Ele permite que programadores, utilitários ou qualquer usuário cadastrado na plataforma contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo. É amplamente utilizado por programadores para divulgação de seus trabalhos ou para que outros programadores contribuam com o projeto, além de promover fácil comunicação através de recursos que relatam problemas ou mesclam repositórios remotos.

Por meio da API Graphql é possível minerar informações sobre os repositórios contidos na plataforma. Este trabalho vem com o objetivo analisar os dados obtidos dos 1000 repositórios mais populares, classificados a partir dos seus números de estrelas, assim relacionando suas características será possível responder as seguintes perguntas:

a) Sistemas populares são maduros/antigos?
b) Sistemas populares recebem muita contribuição externa?
c) Sistemas populares lançam *releases* com frequência?
d) Sistemas populares são atualizados com frequência?
e) Sistemas populares são escritos nas linguagens mais populares?
f) Sistemas populares possuem um alto percentual de issues fechadas?
g) Sistemas escritos em linguagens mais populares recebem mais contribuição externa, lançam mais releases e são atualizados com mais frequência?

Para cada pergunta foi elaborado hipóteses que serão testadas com os resultados obtidos.

a) Como a plataforma foi fundada em 2008 e o processo de adquirir uma alta avaliação pode demorar conforme um projeto ganhe popularidade, então é possível que os repositórios mais populares tenham uma maturidade acima de 5 anos (1825 dias).
b) Se um sistema é popular ele consequentemente deve possuir um maior número de pessoas interessadas em contribuir com seu desenvolvimento, então eles devem sim receber um maior número de contribuições externas.

c) Caso a segunda hipótese esteja correta se um projeto recebe um maior número de contribuições então penso que seu número de releases deva ser maior para comportar o número de modificações feitas pelos desenvolvedores de todo mundo.
d) Isso se correlaciona com as duas últimas hipóteses, se o sistema recebe maiores contribuições e lança um maior número de releases, então este deveria ser atualizado com uma maior frequência.
e) Suponho que um dos motivos que possa fazer um repositório ser muito popular é sua linguagem de programação, pois assim aumenta o número de possíveis desenvolvedores ao projeto, dessa forma acredito que estes sistemas em sua maioria devem utilizar as linguagens mais populares.
f) Para um sistema ser popular é importante o engajamento da comunidade, com isso aumentaria a agilidade na solução de erros reportados, penso que estes sistemas devem possuir um número alto de issues resolvidas(fechadas).
g) Caso a quinta hipótese esteja correta é possível supor que os sistemas que utilizam as linguagens mais populares recebam um maior número de contribuições visto o maior número de possíveis colaboradores.

2. Metodologia

Para mineração dos dados foi utilizado a API Graphql em conjunto com script desenvolvido na linguagem Python, retornando os primeiros 1000 repositórios no GitHub com mais de 100 estrelas em sua avaliação. Na análise deste trabalho se utilizará com data de coleta dos dados em 03/06/2020 e todos estão disponíveis no repositório do projeto. Para responder as perguntas foi utilizado as seguintes métricas:

a) A idade do repositório foi calculada a partir da data de obtenção dos dados subtraindo a data de sua criação(*createdAt*).
b) Número de contribuições será o total de pull requests aceitas(*pullRequests*).
c) Frequência de releases será o total de releases(*releases*).
d) O tempo desde a última atualização do repositório foi calculada a partir da data de obtenção dos dados subtraindo a data da última atualização(*updatedAt*).
e) Por meio do campo "*primaryLanguage*" se obteve a linguem de cada repositório, juntando os resultados tivemos as linguagens mais utilizas.
f) O porcentual foi obtido pela razão entre número de issues fechadas(*closedIssues*) pelo total de issues (*totalIssues*).

g) Se escolheu as 10 linguagens com maior número de repositórios e foram refeitos os cálculos dos itens 2, 3 e 4. Após isso foi subtraído do total de todos os repositórios se obtendo os dados para as linguagens menos populares.

3. Resultados Obtidos

a) Analisando os dados da tabela abaixo é possível perceber que mais de 86,4% dos repositórios possuem idade maior que 2 anos e apenas 16,2% foram criados a mais de 8 anos. A partir da mediana se tem o valor de 1974.5 dias.

Intervalo de Idade(dias)	Número de Repositórios
x<1000	136
1000<x< 3000	702
x>3000	162

b) Por meio dos resultados é possível perceber que somente 217 repositórios possuem mais de 1000 pull requests e 78,3% possuem menos de 1000 *pull requests*. A partir da mediana se tem o valor de 281 *pull requests*.

Intervalo Pull Requestes	Número de Repositórios
x<100	294
100<x<500	352
500<x<1000	137
1000<x<10000	191
x>10000	26

c) Com dados obtidos é possível perceber que 88,2% dos sistemas possuem menos de 100 *releases*. A partir da mediana se tem o valor de 7 *releases*.

Intervalo releases	Número de Repositórios
x<50	774
50<x<100	108
100<x<500	114
x>500	4

d) A tabela abaixo mostra que 98,5% dos repositórios possuem menos de 1 dia. A partir da mediana se tem o valor de 0 dias.

Tempo de atualização(dias)	Número de Repositórios
x=0	985
x>0	15

e) Os resultados abaixo foram comparados com as 25 linguagens mais populares, destas 19 linguagens foram identificadas, totalizando 805 repositórios.

Linguagem	Repositórios
JavaScript	302
Python	94
Java	70
Go	59
TypeScript	48
C++	45
CSS	25
C	23
Swift	23
HTML	22
Shell	22
PHP	19
Ruby	17
Objective-C	12
Kotlin	11
C#	8
Assembly	2
Scala	2
Perl	1
Total	805
Linguagens não identificada	116
Linguagens não populares	79

f) Obteve-se o valor de 0.864478 ou 86,64% de *closed issues* por *total issues*

g) Baseando-se nas 19 linguagens mais populares identificadas obteve-se a tabela abaixo:

	Linguagens Populares	Linguagens Não Populares
Total Pull Requests	338.5	122
Total Releases	14	0
Tempo de Atualização	0	0

4. Discussão

Após compara com os resultados obtidos com a hipóteses feitas no início, pode-se concluir que:

a) Maior parte dos repositórios apresentaram tempo de criação maior que cinco anos, foi o esperado a partir da hipótese, de que repositórios populares possuem uma maior maturidade.

b) Os repositórios apresentaram baixa quantidade de *pull requests*, ao contrário da hipótese, o número de contribuições dos usuários não está relacionado com a popularidade do projeto.

c) Assim como os *pull requests,* os repositórios apresentaram um baixo número de releases, indo contra a hipótese feita, o número de incrementos ao sistema não está relacionado com a popularidade do projeto.

d) Mesmo com a baixa de *pull requests* e *releases*, os sistemas apresentaram uma alta taxa de atualização em sua maioria, indo de acordo com a hipótese feita, repositórios populares possuem uma alta frequência de atualização.

e) Mais de 80% dos repositórios apresentaram o uso de linguagens populares, indo de acordo com a hipótese feita, repositórios populares utilizam linguagens populares.

f) Foi apresentado uma alta taxa de *issues* fechadas, indo de acordo com a hipótese feita, repositórios populares possuem alto índice de *closed issues*.

g) Linguagens populares apresentaram uma maior mediana no total de *pull requests* e *releases,* o tempo de atualização foi igual para os dois tipos de linguagens, indo de acordo com a hipótese feita, repositórios que utilizam linguagens populares possuem alto índice de *pull request e releases*.