Laboratório #01 - Características de Sistemas Populares GitHub

Gabriel Santana Barroso

(gsbarroso@sga.pucminas.br)

Instituto de Ciências Exatas e Informática

Pontifícia Universidade Católica de Minas Gerais (PUC-MG)

Introdução

O GitHub é uma plataforma popular de versionamento de software, em que seus usuários podem criar, manter e gerenciar seus próprios repositórios de código na nuvem gratuitamente (e também apresenta algumas features pagas). A plataforma garante acesso aos repositórios -- que podem ser públicos ou privados -- por parte de seus usuários a partir de qualquer máquina do mundo, desde que esteja conectada à internet. Possui também funcionalidades como: estatísticas sobre as atividades nos repositórios, participação em repositórios de terceiros, abertura de issue report, realização de pull requests etc. O GitHub também disponibiliza uma API, através da qual é possível obter dados sobre repositórios por meio de queries com parâmetros bem definidos.

Este é um trabalho de medição e experimentação de software, cujo objetivo é analisar algumas características de sistemas populares de código aberto que estão armazenados em repositórios no GitHub. As perguntas que motivam este trabalho estão listadas a seguir:

1) Sistemas populares são maduros/antigos?
2) Sistemas populares recebem muita contribuição externa?
3) Sistemas populares lançam releases com frequência?
4) Sistemas populares são atualizados com frequência?

5) Sistemas populares são escritos nas linguagens mais populares do mercado?
6) Sistemas populares possuem um alto percentual de issues fechadas?
7) Sistemas escritos em linguagens mais populares recebem mais contribuição externa, lançam mais releases e são atualizados com mais frequência?

Foram elaboradas hipóteses para cada uma dessas questões, a serem testadas por esta pesquisa:

1) Hipótese: um sistema, para ser considerado maduro, deve ter pelo menos 730 dias (2 anos) de idade. O fenômeno de popularização de um sistema leva tempo, tendo em vista que se faz necessária divulgação e boa avaliação (alto número de estrelas no GitHub) do software, o que requer que este software seja de boa qualidade. Espera-se, portanto, que sistemas populares sejam maduros, uma vez que o processo de reconhecimento e ganho de credibilidade é moroso.
2) Hipótese: tomando exemplos como Linux, VueJS, TensorFlow e NodeJS, que são tecnologias de código aberto populares no mercado, muito bem avaliadas no GitHub e que têm grande engajamento da comunidade em prol de sua melhoria constante, é esperado que sistemas populares recebam muita contribuição externa, sim.
3) Hipótese: aqui, devem ser ponderadas as duas hipóteses anteriores -- por um lado, se um sistema recebe muita contribuição externa, espera-se que seu número de releases aumente, mas por outro, faz parte da maturidade do software ele ter como uma de suas principais características a estabilidade. Espera-se um número considerável de releases, mas não muito elevado.

4) Hipótese: levando em consideração a hipótese levantada para a questão 2, espera-se que softwares populares sejam atualizados com frequência, graças ao engajamento de suas comunidades.
5) Hipótese: espera-se que sim. Um dos fatores de influência na popularidade de um software é a linguagem em que ele foi desenvolvido, pois, inevitavelmente, isto acaba atraindo a atenção e o engajamentos de desenvolvedores que trabalham com a linguagem, em algum momento. E se a linguagem é popular, a base de desenvolvedores é ampla -- o que provoca a difusão e a boa avaliação do software por parte destes, quando o software é de qualidade.
6) Hipótese: espera-se que sim. A correção de problemas encontrados e reportados e a evolução do código são fatores vitais para a popularidade de um sistema. Não espera-se que, necessariamente, um software popular tenha muitas issues reportadas, mas espera-se que dentre as reportadas, o índice de issues fechadas (ou seja, resolvidas) seja alto.
7) Hipótese: levando em consideração as hipóteses anteriores, espera-se que sim.

Metodologia

Esta é uma pesquisa de cunho descritivo que realiza uma abordagem quantitativa. Foram minerados exatamente 1000 repositórios de software no GitHub (os mais bem avaliados) para análise, uma vez elaboradas as hipóteses. Foi elaborado um script Python com a finalidade de extrair e tratar os dados necessários via API do GitHub. Todos os artefatos gerados acompanham este documento num zip e podem ser encontrados em: https://github.com/g-santana/GitHubExplorer-ex1.

Resultados obtidos

Seguem as respostas encontradas para cada uma das perguntas feitas neste trabalho:

1) Métrica utilizada: idade do repositório (em dias). Mediana de idade dos repositórios pesquisados: 1861 dias.
2) Métrica utilizada: número de pull requests aceitas. Mediana de pull requests aceitas dos repositórios pesquisados: 245 pull requests aceitas.
3) Métrica utilizada: número de releases do software. Mediana de releases: 6 releases.
4) Métrica utilizada: tempo (em dias) desde a última atualização feita nos repositórios. Mediana de dias corridos desde a última atualização: 0 dias. A data de última atualização dos repositórios coincidiu com o dia do levantamento destes dados (9 de setembro de 2019). Em decorrência disso, não foi possível obter um resultado claro para esta pergunta.
5) Métrica utilizada: número total de repositórios desenvolvidos (primariamente) nas 10 linguagens de programação mais populares segundo os dados apresentados em: https://www.devmedia.com.br/top-10-linguagens-de-programacao-mais-usadas-no-mercado/39635. Total: 648 repositórios dentre os 1000 (64,8%).
6) Métrica utilizada: razão entre o número de issues fechadas e o número total de issues. Mediana da razão: 0.85 (85%).
7) Métrica utilizada: comparação de repositórios cuja linguagem primária é uma linguagem popular com os demais repositórios, segundo os parâmetros da pergunta 7. Respectivos resultados:
    i) Mediana de pull requests aceitas: 306 contra 164
    ii) Mediana de releases: 12 contra 0
    iii) Mediana de dias corridos desde a última atualização: 0 contra 0

Conclusão

Comparando as hipóteses com os resultados, pode-se concluir que:

1) Repositórios populares são maduros.
2) Repositórios populares recebem uma taxa considerável de contribuição externa.
3) Repositórios populares têm poucas releases.
4) Não foi possível avaliar a frequência de atualização dos repositórios.
5) Mais de metade dos sistemas populares estão escritos em linguagens de programação populares, o que é uma parcela muito significativa.
6) Repositórios populares têm um alto índice de issues fechadas.
7) Sistemas desenvolvidos em linguagens populares recebem mais contribuição externa e lançam mais releases que os outros. Não foi possível avaliá-los quanto a frequência de atualização.