PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS E INFORMÁTICA
UNIDADE EDUCACIONAL PRAÇA DA LIBERDADE
Bacharelado em Engenharia de Software

Kelton Melo de Oliveira Fonseca

Análise Comparativa de Repositórios Python e JAVA

Belo Horizonte
2020

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS E INFORMÁTICA
UNIDADE EDUCACIONAL PRAÇA DA LIBERDADE
Bacharelado em Engenharia de Software

Kelton Melo de Oliveira Fonseca

Trabalho de Software apresentado como requisito parcial à aprovação na disciplina Laboratório de Experimentação de Software

Professor: Humberto Torres Marques Neto

Belo Horizonte
2020

1. Proposta

Objetivo: Analisar a qualidade de repositórios desenvolvidos na linguagem Python comparando-os com repositórios desenvolvidos na linguagem JAVA sob a perspectiva de características levantadas através de ferramentas de análise de código.

Questões e métricas:

- Quais as características dos top-100 repositórios Java mais populares?
- Quais as características dos top-100 repositórios Python mais populares?
- Repositórios Java e Python populares possuem características de “boa qualidade” semelhantes?
- A popularidade influencia nas características dos repositórios Java e Python?

Utilizaremos como fatores de qualidade métricas associadas à quatro dimensões:

- Popularidade: número de estrelas, número de watchers, número de forks dos repositórios coletados
- Tamanho: linhas de código (LOC e SLOC) e linhas de comentários
- Atividade: número de releases, frequência de publicação de releases (número de releases / dias)
- Maturidade: idade (em anos) de cada repositório coletado

2. Introdução e hipóteses

Este trabalho visa analisar os 100 repositórios Java mais populares do GitHub, juntamente aos 100 repositórios Python mais populares e fazer uma comparação entre os trabalhos desenvolvidos nessas linguagens. Ele comparará tanto características que independem da linguagem, como quantidade de releases, watchers e forks, mas também a questão de números de linha de código.

Espera-se que os repositórios Java tenham um grande número de watchers e de forks, pois Java se encontra no topo dos rankings de linguagens populares, então o número de usuários que clonam o repositório pode ser grande, mas ainda assim menor que o número de usuários que querem receber atualizações desses repositórios, para ficarem a par de correções de erros e de possibilidade de novos releases. Por Java ser uma linguagem de programação que exige muitas linhas ao se criar classes, importar bibliotecas, deverá haver projetos com uma quantidade grande de linhas de código, alguns projetos devem ser bem maduros, pois já estamos na versão 15 do Java. Mas a quantidade de releases deve se manter semelhante à vista em trabalhos anteriores, com 1 release a cada 2 meses por projeto.

Enquanto isso, os repositórios Python terão menos watchers e forks e essa característica se refletirá no número de releases. Com menos clones do repositórios e menos usuários querendo saber dos repositórios pode ser que o número de releases diminua. Sua sintaxe é mais simples, o que gerara códigos populares com menos SLOCs, e acredito que códigos menores terão maior

número de estrelas. Quanto a idade, repositórios Python devem ser tão maduros quanto repositórios Java, se não forem mais velhos, pois a linguagem é mais antiga.

Quanto a boa qualidade, estaremos analisando apenas o número de linhas de código neste trabalho, porém acredito que essa característica não seja semelhante nos repositórios Java e Python devido ao fato de essas linguagens terem sintaxes muito distintas. Enquanto Python é uma linguagem interpretada, Java é uma linguagem compilada necessariamente orientada a objetos, e isso acrescenta algumas linhas de código.

Acredito que a popularidade influencia sim nas características de qualidade do repositório, no caso do Python, repositórios com menor numéros de SLOCs devem ser mais populares, pois é uma linguagem desenvolvida para ser simples. Porém em Java, a economia de linhas pode dificultar a leitura e diminuir a popularidade dos repositórios, dessa forma espera-se que códigos mais detalhados e comentados, com maior número de LOCs sejam códigos mais populares.

3. Metodologia e Resultados

Serão geradas querys que serão enviadas para API GraphQL do GitHub através de um programa simples desenvolvido em Python de forma a tentar analisar algumas das métricas propostas as métricas propostas

Abaixo é possível ver as duas principais queries executadas:

```
Query Python { search(query: "language:python", type: REPOSITORY, first: 100) {
    edges {
        node {
            ... on Repository {
                nameWithOwner
                createdAt
                stargazerCount
                primaryLanguage { name }
                watchers {  totalCount  }
                forks {  totalCount   }
                releases { totalCount   }
            }
        }
    }
```

```
        repositoryCount
}}

Query Java { search(query: "language:java", type: REPOSITORY, first: 100) {
        edges {
                node {
                        ... on Repository {
                                nameWithOwner
                                createdAt
                                stargazerCount
                                primaryLanguage { name }
                                watchers {  totalCount  }
                                forks {  totalCount   }
                                releases { totalCount   }
                        }
                }
        }
        repositoryCount
}}
```

Foram necessárias duas queries, uma para fazer a requisição dos repositórios Python e uma para os repositórios Java. Contudo não era possível capturar nessas queries o números de linhas de código de cada repositórios e para tal foi usada uma biblioteca do Node.js, chamada SLOC que apresenta número de linhas totais, linhas de código fonte, linhas vazias, quantos são os blocos de comentário e quais são linhas simples de comentário.

Para cada repositório foi feito seu download, e então analisado pela SLOC e as linhas dadas como físicas foram consideradas o total do repositórios, enquanto os dados de fonte foram as linhas de código fonte. Foi feita a subtração do total pelas linhas de código fonte para se obter linhas de comentários e vazias.

Os dados a seguir podem ser encontrados nas planilhas dispostas no link: https://github.com/notlekmelo/lab36P

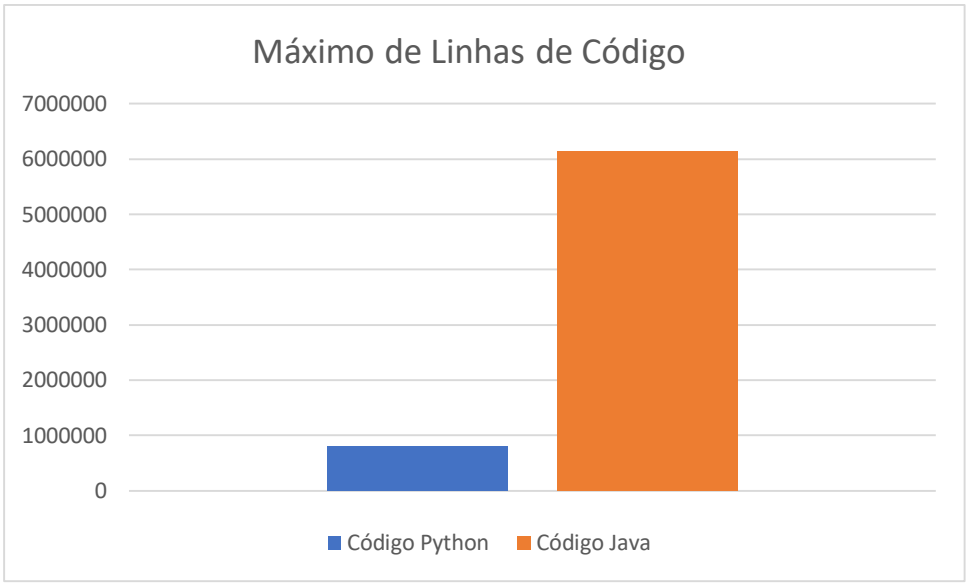
4. Análise dos resultados

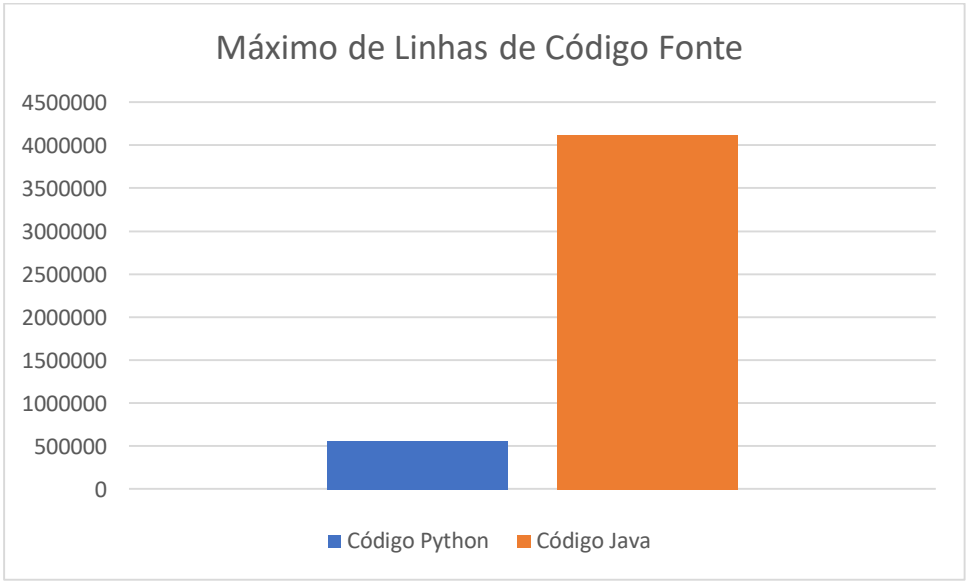
Com a execução do programa Python foi gerado um arquivo .csv que permitiu que a análise dos dados fosse feita por formulas em uma planilha Excel. Tal análise não pode ser dita como totalmente confiável já que alguns dos dados requisitados vieram com valores nulos ou inválidos em determinados projetos, hipótese que é descartada devido ao tamanho dos projetos buscados.

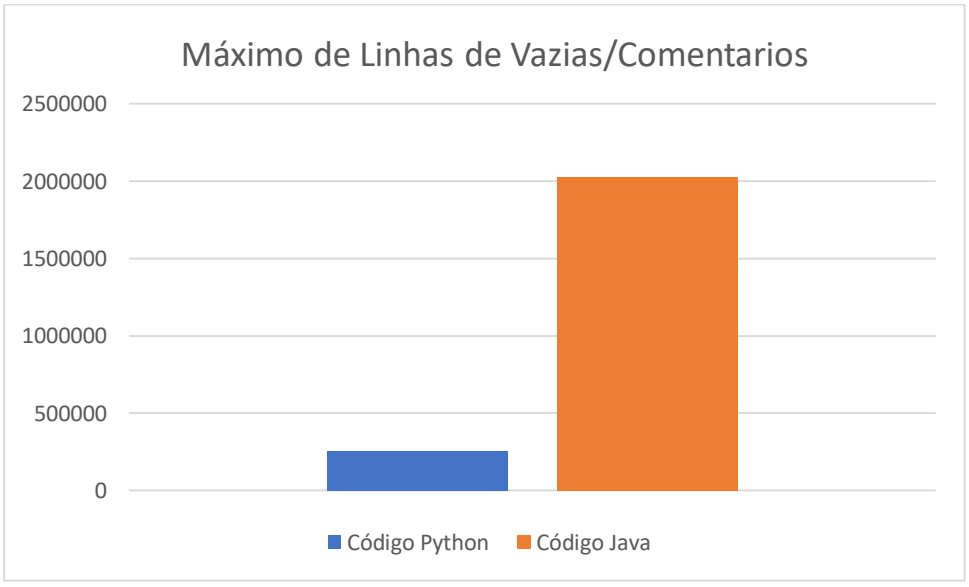
Os repositórios Java tem uma mediana de idade de 5 anos, com o mais novo tendo sido criado ainda em 2020 e o mais velho em 2008. Quanto à popularidade o máximo de estrelas que possuem é 112.027 este repositório possui 34.893 watchers e 5.226 forks, mas a mediana de estrelas, watchers, e forks é de 14.394,50, 3.918 e 768,50. Esse repositório não possui muitas linhas de código, sendo que a mediana de linhas de código é de 50.529 e das linhas de código fonte são 34.235,5 nesse caso foi completamente inviável utilizar-se da média visto que houve um valor de 58 linhas de código fonte em dissonância com um repositório com 4.119.411 linhas de código fonte. Quanto aos releases, o máximo de releases é de 191, de um repositório de 4 anos, mas a mediana é de 25 e dos releases por dia é de 0,012425021 o que significa 1 release a cada aproximadamente 80 dias, ou 1 release a cada dois meses e meio.

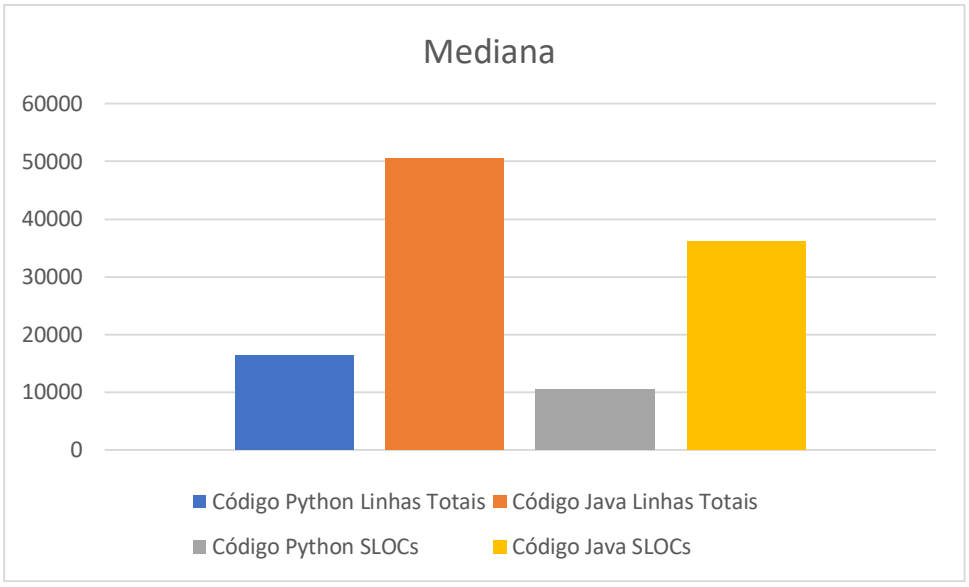
Ao analisarmos os repositórios Python percebemos que quando se trata de sua idade, não poderemos analisar a média, pois o desvio padrão foi de 2,89, então sabemos que a mediana de idade destes repositórios é de 5 anos, sendo que o mais velho possui 12 anos de criação e o mais novo foi criado ainda no ano de 2020. O repositório mais popular possui 3 anos de idade e 108.455 estrelas, mas ele também é destoante da média de estrelas dos repositórios Python, e bem acima da mediana de 12.299,50 estrelas. Quanto a mediana dos números de watchers e de forks temos 1.955,50 e 493,50 respectivamente números que refletem quase a metade do que foi visto nos repositórios Java, e que se reflete nas releases, com uma mediana de 12 releases nesses repositórios, o que gera uma mediana de 0,007063017 releases por dia, ou seja 1 release a cada aproximadamente 5 meses. Sendo que o repositório que possui releases mais frequentes tem uma mediana de 0,082692308 releases/ dia, demorando cerca de 12 dias para lançar um release. Por fim, considerando-se a quantidade de linhas de código, repositórios Python possuem uma mediana de 16.486 com máximo de 818.195 linhas totais de código e mínimo de 7. Contudo há também as linhas de comentário que se retiradas desse cálculo mudam os valores de mediana, máximo e mínimo para 10.419, 560.677 e 6 respectivamente.

Dessa forma podemos ver que ao se comparar características de boa qualidade com relação a tamanho do software temos os seguintes resultados


Máximo de Linhas de Código
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
Código Python
Código Java


Máximo de Linhas de Código Fonte
4500000
4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
Código Python
Código Java


Máximo de Linhas de Vazias/Comentarios
2500000
2000000
1500000
1000000
500000
0
Código Python
Código Java



Assim percebemos que a popularidade influenciou pouco nas características dos repositórios.