Caracterização de bibliotecas de UI

Aylton B. de Almeida Junior[1], **Lucca Romaniello Benjamin**[2], **Pedro Paulo Andrade**[3]

[1]Engenharia de Software – Pontifícia Universiade Catolica de Minas Gerais (PUC MG)
Belo Horizonte – MG – Brazil

{abajunior, lbenjamin, pedro.faria.1115474}@sga.pucminas.br

1. Introdução

”Nos últimos anos, tornar as coisas mais utilizáveis se tornou uma responsabilidade de quase todos. Agora, designers visuais e desenvolvedores costumam fazer coisas como design de interação (decidir o que acontece a seguir quando o usuário clica, toca ou desliza) e arquitetura de informações (descobrir como tudo deve ser organizado)”. [Krug 2014].

Com base na citação de Steve Krug, percebemos que as interfaces visuais no mundo digital se tornam amplamente necessárias, especialmente para possiblitar que usuários completem das mais diversas tarefas em produtos digitais. Com o avanço da tecnologia e a crescente demanda por desenvolvimento de sistemas amigáveis aos usuários finais, bibliotecas que auxiliam a criação de interfaces de usuário se tornam cada vez mais frequentes no ramo do desenvolvimento.

Dessa forma, com o crescimento na utilização de frameworks web e de bibliotecas de UI *(User Interface)*, surgem alguns questionamentos em relação à tais sistemas, como por exemplo:

1. As bibliotecas mais populares são mais abertas a novos colaboradores?
2. Bibliotecas baseadas nas linguagens mais populares possuem uma menor curva de aprendizado?
3. As bibliotecas de UI se tornaram necessárias para os projetos atuais?

Será realizado um experimento de medição de software para tentar responder os questionamentos apontados acima e, inicialmente, sua metodologia será descrita a seguir.

2. Metodologia

Para a realização da pesquisa será necessário recuperar dois grupos principais de dados, sendo o primeiro necessário para conseguir fazer a coleta do segundo. O primeiro grupo de dados é uma lista de 1000 repositórios de bibliotecas de componentes para construção de interfaces gráficas, utilizando de tópicos do Github para filtra-los. Já o segundo se trata de 1000 repositórios NodeJs, de forma que usaremos o arquivo *package.json* do repositório para verificar se eles usam ou não alguma biblioteca de UI.

2.1. Sobre a Questão 1

A primeira métrica da questão 1 é a relação entre o número de *Pull Requests* feitos por novos contribuidores pelo número total de *PRs*. Para isso seram recuperados todos os *Pull Requests* e seus autores, caso um autor possua apenas um *PR* ele é um considerado novo autor. Já a segunda métrica é o tempo médio para aceitação de *Pull Requests* feitos

por novos contribuidores, esses que serão caracterizados da mesma forma que na métrica 1.

Ambas essas métricas são acessíveis por meio da API GraphQL do Github. Dessa forma um *script* em Python será escrito, utilizando de cursores para ultrapassar o limite de 100 repositórios por *request*. Esse código tem como objetivo gerar um arquivo csv contendo uma sumarização dos dados para analise posterior.

2.2. Sobre a Questão 2

Primeiramente, assumimos que perguntas realizadas no site *StackOverflow* associadas à tais linguagens possuem um percentual de respostas maior do que aquelas menos populares. Dessa forma, utilizando o *Stack Exchange Data Explorer*, uma interface de consultas SQL na base dos sites pertencentes a essa stack (nesse caso, incluindo o *StackOverflow*), será calculado o percentual de questões respondidas para as bibliotecas analisadas.

Para complementar a formulação da resposta para a hipótese inicial, também será calculada a frequência de atualização de repositórios que utilizam as bibliotecas baseadas nas linguagens mais populares. Essa métrica se baseia na suposição que bibliotecas populares são atualizadas mais frequentemente e, dessa forma, se torna mais complexo manter o conhecimento relacionado àquela biblioteca atualizado. Para isso, será desenvolvido um programa em *Python* que consulta a API GraphQL da empresa GitHub, esta que é provavelmente uma das principais plataformas de hospedagem de código fonte do mundo e que também armazena as principais bibliotecas *open source* atualmente.

2.3. Sobre a Questão 3

Inicialmente será feita uma busca utilizando a API GraphQL do Github para encontrar todos os repositórios criados no ano de 2020.Em seguida um script desenvolvido em Python utilizando a biblioteca GitPy vai buscar os repositórios, checar se existe o arquivo ”Package.json” e caso haja este arquivo, irá buscar a existência de algum nome dos repositórios encontrados o processo da Questão 1 e 2.

Dessa forma será criado um filtro onde dividirá em duas frentes: repositórios que possuem bibliotecas de user interface e repositórios que não utilizam.

2.4. Conclusão

Sendo assim após toda a operação de recuperar os repositórios de bibliotecas relacionadas a UI do Github, buscar os repositórios criados em 2020 e filtrar todos estes dados entre os que utilizam alguma das bibliotecas e os que não utilizam de nenhuma biblioteca teremos dados suficientes para definir se o uso de bibliotecas UI se tornou uma característica relevante durante as criações de novos projetos.

3. Referências

Referências bibliográficas utilizadas pelo projeto: [Daniel et al. 2007], [Wikipedia], [Desolda et al. 2017], [Balland et al. 2013], [Org], [Inc. 2020], [Pfeiffer et al. 2016], [Krug 2014], [W. Wang 2020], [Sánchez et al. 2020], [Alafaireet 2006], [Badre and Jacobs 1999], [Khan et al. 2011], [Vinnakota 2016], [Erdős 2019]

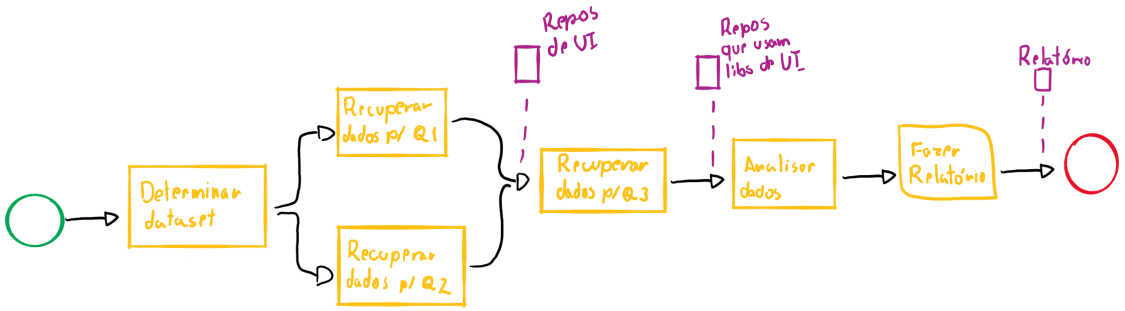

Figure 1. Imagem metodologia

References

Alafaireet, P. (2006). Graphic user interface: Needed design characteristics for successful physician use. In *2006 ITI 4th International Conference on Information Communications Technology*, pages 1–1.

Badre, A. and Jacobs, A. (1999). Usability, aesthetics, and efficiency: an evaluation in a multimedia environment. In *Proceedings IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 103–106 vol.1.

Balland, E., Consel, C., N'Kaoua, B., and Sauzéon, H. (2013). A case for human-driven software development. In *2013 35th International Conference on Software Engineering (ICSE)*, pages 1229–1232.

Daniel, F., Yu, J., Benatallah, B., Casati, F., Matera, M., and Saint-Paul, R. (2007). Understanding ui integration: A survey of problems, technologies, and opportunities. *IEEE Internet Computing*, 11(3):59–66.

Desolda, G., Ardito, C., Costabile, M. F., and Matera, M. (2017). End-user composition of interactive applications through actionable ui components. *Journal of Visual Languages Computing*, 42:46–59.

Erdős, F. (2019). Economical aspects of ux design and development. In *2019 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, pages 211–214.

Inc., G. (2020). The state of the octoverse.

Khan, M., Sulaiman, S., Said, A. M., and Tahir, M. (2011). Classification of usability issues for haptic systems. In *2011 7th International Conference on Emerging Technologies*, pages 1–4.

Krug, S. (2014). *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. New Riders Publishing;.

Org, M. Material ui.

Pfeiffer, T., Hellmers, J., Schön, E., and Thomaschewski, J. (2016). Empowering user interfaces for industrie 4.0. *Proceedings of the IEEE*, 104(5):986–996.

Sánchez, V. R., Ayuso, P. N., Galindo, J. A., and Benavides, D. (2020). Open source adoption factors—a systematic literature review. *IEEE Access*, 8:94594–94609.

Vinnakota, T. (2016). A conceptual framework for complex system design and design management. In *2016 Annual IEEE Systems Conference (SysCon)*, pages 1–6.

W. Wang, J. Cheng, J. L. C. G. (2020). How do open source software contributors perceive and address usability? valued factors, practices, and challenges. *IEEE Software*, pages 0–0.

Wikipedia, a. e. l. Component-based software engineering.