UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

MATEUS VIEIRA TORRES

UMA INVESTIGAÇÃO SOBRE A INFLUÊNCIA DA PRÁTICA DE ATRIBUIÇÃO DE RÓTULOS ESPECÍFICOS PARA NOVATOS EM REPOSITÓRIOS DE CÓDIGO ABERTO

CAMPO MOURÃO

2021

MATEUS VIEIRA TORRES

UMA INVESTIGAÇÃO SOBRE A INFLUÊNCIA DA PRÁTICA DE ATRIBUIÇÃO DE RÓTULOS ESPECÍFICOS PARA NOVATOS EM REPOSITÓRIOS DE CÓDIGO ABERTO

A Research on the Influence of Newcomer Label Assignment Practice in Open Source Repositories

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Ciência da Computação do Curso de Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Igor Scaliante Wiese

CAMPO MOURÃO

2021

MATEUS VIEIRA TORRES

UMA INVESTIGAÇÃO SOBRE A INFLUÊNCIA DA PRÁTICA DE ATRIBUIÇÃO DE RÓTULOS ESPECÍFICOS PARA NOVATOS EM REPOSITÓRIOS DE CÓDIGO ABERTO

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Ciência da Computação do Curso de Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Data de aprovação: 26/novembro/2021

Prof. Dr. Igor Scaliante Wiese
Doutorado
UTFPR

Prof. Dr. André Luis Schwerz
Doutorado
UTFPR

Prof. Dr. Igor Steinmacher
Doutorado
UTFPR

CAMPO MOURÃO

2021

RESUMO

Contexto: Contribuir para projetos de código aberto pela primeira vez pode ser desafiador. Algumas vezes, novatos acabam desistindo de ingressar nesses projetos devido a dificuldade para encontrar uma tarefa apropriada. Com o intuito de atrair novos contribuidores, algumas iniciativas vêm listando repositórios de código aberto, hospedados no GitHub, que praticam a atribuição de rótulos para indicar que uma tarefa é adequada para novatos. Sabendo que projetos de software livre muitas vezes dependem de contribuidores voluntários, práticas com potencial para atraí-los merecem atenção. Atualmente, iniciativas como o *Up-For-Grabs* assumem que a rotulação de tarefas indicadas para novatos é uma prática que influencia positivamente na atratividade de novatos em projetos de código aberto. Entretanto, até o momento existem poucas evidências documentadas para sustentar esta hipótese.

Objetivo: Em vista disso, o objetivo desse trabalho consiste em investigar se a prática de rotular tarefas como "indicadas para a primeira contribuição" de fato aumenta a atratividade de novatos em projetos de software livre que a empregam em seus repositórios.

Método: Para verificar se a hipótese supracitada é verdadeira, foram executadas análises comparativas por meio de testes estatísticos não paramétricos aplicados a um conjunto de 1060 projetos de 10 diferentes linguagens e cujo código é aberto e disponível no GitHub. Das amostras coletadas foram comparados estatisticamente as distribuições referentes ao ingresso semanal de novatos antes e depois da adesão da prática. Complementarmente também foram comparadas as distribuições referente a quantidade total de novatos dos projetos adotantes da prática com os não adotantes.

Resultados: Fora descoberto que existe uma vasta variação de termos sendo utilizados pela comunidade como rótulos para novatos e esses termos nem sempre são intuitivos. Também percebeu-se que uma parcela significante dos projetos vem utilizando a rotulação para indicar tarefas recomendadas para novatos. Testes estatísticos apontaram que 63% dos projetos adotantes da prática apresentaram um aumento significativo no ingresso semanal de novatos após a adoção da prática. Os testes também evidenciaram que projetos praticantes apresentam uma quantidade maior de novatos do que projetos não praticantes.

Conclusões: Os resultados obtidos nesta pesquisa evidenciaram que a utilização de rótulos para indicar tarefas para novatos pode aumentar significativamente a atratividade de novatos em projetos de código aberto. Entretanto, ainda existe margem para aprimoramento da prática, principalmente no que se refere aos termos utilizados nos rótulos para novatos.

Palavras-chaves: Software Livre. Código Aberto. Recomendação de Tarefas. Novatos. *Up-For-Grabs*.

ABSTRACT

Context: Contribute to open source projects for the first time can be challenging. Sometimes, newcomers end up giving up of joining these projects because of the difficulty in finding a task that they feel comfortable to contribute. Aiming to attract new contributors, some projects like *Up-For-Grabs* are listing open source repositories, hosted on GitHub, that practice assigningment of specific labels to indicate that a task is recommended for picking by newcomers. Since free software are maintained oftentimes by their contributors, practices that can lead to attraction of new contributors are more than welcome and deserve attention. Currently, initiatives such as *Up-For-Grabs*, assume that the labeling of tasks recommended for newcomers is a practice that can influence positively the attractiveness of newcomers into open source projects. However, to date there is little documented evidence to support this hypothesis.

Objective: In view of this, the objective of this work is to investigate if the labeling of tasks as "recommended for first contribution" actually increases the attractiveness of newcomers to the free software projects that use it in their repository.

Method: To verify whether the above-mentioned hypothesis is true, comparative analysis was performed through non-parametric statistical tests applied to a set of 1060 projects out of 10 different languages and whose code is open source and available on GitHub. Of the collected samples, the distributions referring to the weekly ingress of newcomers before and after the adoption of practice were statistically compared. Complementarily, the distributions referring to the total number of newcomers from projects adopting the practice were also compared with non-adopters.

Results: It had been discovered that there is a wide range of terms being used by the community as labels for newcomers and these terms are not always intuitive. It was also noticed that a significant portion of projects have been using labeling to indicate recommended tasks for newcomers. Statistical tests showed that 63% of the projects adopting the practice had a significant increase in the weekly entry of newcomers after adopting the practice. The tests also showed that projects who use newcomers labels have a greater number of newcomers than projects who don't.

Conclusions: The results obtained in this research showed that the use of labels to indicate tasks for newcomers can significantly increase the attractiveness of newcomers in open source projects. However, there is still room for improvement in the practice, especially regarding to the terms used in labels for newcomers.

Keywords: Free Software. Open Source. Tasks Recommendation. Newcomers. *Up-For-Grabs*.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API: *Application Programming Interface.* 34

FSF: *Free Software Foundation.* 9

GSoC: *Google Summer of Code.* 7, 15

HTTP: *Hypertext Transfer Protocol.* 34

JSON: *JavaScript Object Notation.* 34, 35

MWW: *Mann-Whitney-Wilcoxon.* 22, 29–32, 37, 41–43

OSI: *Open Source Initiative.* 9

SL: Software Livre. 7–15, 17–22, 31, 45

SO: Sistema Operacional. 10

URL: *Uniform Resource Locator*. 19, 26

SUMÁRIO

1 INTRODUÇÃO

Software livre é o software que concede liberdade ao usuário para executar, acessar e modificar o código fonte, e redistribuir cópias com ou sem modificações. Sabe-se que projetos de Software Livre (SL) vêm ganhando cada vez mais espaço no mercado de software. SL é um termo utilizado para denominar programas de computador cuja licença permite a execução, cópia, modificação e redistribuição de forma gratuita (FSF, 2004).

Projetos de SL normalmente contam com a colaboração da vasta comunidade de desenvolvedores espalhados pelo mundo todo para sua sustentabilidade. Indivíduos que contribuem para projetos de SL não necessariamente são empregados ou pagos por instituições, as contribuições que realizam podem ser simplesmente por iniciativa voluntária. Alguns estudos da literatura exploram as motivações por trás de contribuições voluntárias para projetos de SL. Algumas das motivações conhecidas são: identificação com o projeto, necessidade pessoal, expectativa de retorno no futuro, recompensas externas, aprendizado e altruísmo (HARS, 2002). Silva et al. (2017) por exemplo, explora em seu trabalho a eficiência de iniciativas como programa *Google Summer of Code* (GSoC) que tem como intuito promover contribuições e atrair novos desenvolvedores para as comunidades de SL. E também Hertel et al. (2003) que investigou por meio de um questionário online o que motivou 141 desenvolvedores a contribuírem para um dos maiores projetos de SL existentes, o kernel do Linux.

Quando desenvolvedores propõem-se a realizar sua primeira contribuição para projetos de SL, muitas vezes não estão cientes da complexidade envolvida no processo até ter uma contribuição aceita. Em seu estudo, Steinmacher et al. (2014) mapeiam as principais barreiras enfrentadas por novatos ao tentarem ingressar em um projeto. Uma das barreiras identificadas é a dificuldade que novatos têm de encontrar uma tarefa adequada para contribuir pela primeira vez. Essa barreira induz muitos indivíduos a desistirem de submeter uma contribuição.

Deste modo, se a sustentabilidade destes projetos beneficia-se de sua atratividade em relação a novos contribuidores, é necessário elaborar e entender práticas com potencial para ampliá-la (SANTOS et al., 2013; YU et al., 2016). Grande parte dos projetos de SL hospedam seus códigos-fontes em ambientes de codificação colaborativos como o GitHub[1], o que facilita o acesso da comunidade a estes códigos e também auxilia na interação colaborativa entre os mantenedores dos projetos e seus respectivos contribuidores. Entre os diversos serviços oferecidos pelo GitHub, existe o sistema de gerenciamento de tarefas que, além de outras funções, possibilita a atribuição de rótulos a tarefas reportadas em um repositório.

As estratégias propostas por estudos da literatura para contrapor a barreira descrita por Steinmacher et al. (2014), convergem no sentido de que sugerem a utilização de meios que auxiliem os novatos a distinguirem a complexidade de uma tarefa (CUBRANIC et al., 2005; WANG; SARMA, 2011; WOLFF-MARTING et al., 2013). Nesse sentido, a função de rotulação é uma das abordagens

[1] https://github.com

que vêm sendo utilizada em projetos de SL para dar suporte a novatos em sua busca por uma tarefa adequada para realizar a primeira contribuição.

Atualmente alguns projetos desenvolvidos pela comunidade, como o *Up-For-Grabs*[2], assumem que esta prática de fato tem potencial para atrair novatos aos projetos de SL e, portanto, vêm listando e divulgando repositórios que a adotam.

Este tema também vem atraindo a atenção de diversos pesquisadores, que cada vez mais buscam entender detalhes dessa prática. Boa parte desses estudos investigam as possíveis características que possam distinguir uma tarefa indicada para novato das demais (TAN et al., 2020) e outros propõe até mesmo automatização dessa atividade utilizando recursos como aprendizagem de máquina (ALDERLIESTEN; ZAIDMAN, 2021).

Neste contexto, o objetivo deste trabalho é investigar a influência desta prática na atratividade de novatos em repositórios de código aberto. Para isso foram realizadas análises quantitativas utilizando testes estatísticos aplicados à distribuição do ingresso de novatos em 1060 dos projetos mais populares (mais estrelados) hospedados no GitHub.

Em projetos que utilizam a prática foram comparadas as distribuições referentes ao período anterior e posterior a adesão da mesma. Também foram comparados o total de novatos entre projetos que adotam a prática e projetos que não o fazem. Ambas as comparações tem como propósito verificar se é possível constatar diferença estatística entre as distribuições e o tamanho do efeito da prática na atratividade dos projetos, o que pode evidenciar a eficiência ou ineficiência da prática.

A investigação proposta neste trabalho justifica-se pela necessidade de verificar a eficiência dos métodos adotados pela comunidade de SL e sugeridos por estudos da literatura para atração de novatos. Caso os resultados indiquem a ineficiência da prática, o esforço e tempo despendido pelos autores das tarefas na atribuição dos rótulos para novatos e pela comunidade ao divulgar projetos que adotam esta prática vêm sendo em vão. Do contrário, validou-se uma prática que deveria ser divulgada e adotada por projetos de SL interessados em ampliar sua atratividade.

Além disso, apesar do empenho de pesquisadores em buscar métricas que possam fomentar a atratividade de projetos de SL (SANTOS et al., 2013; MEIRELLES et al., 2010), até o momento existem poucos estudos documentados na literatura que buscam entender a relação entre a utilização de rótulos para indicar tarefas adequadas para novatos e a atratividade de um projeto, como proposto neste trabalho.

A estrutura deste trabalho é organizada da seguinte maneira: O Capítulo 2 apresenta a fundamentação teórica, no qual são discutidos os principais assuntos associados a este trabalho. No Capítulo 3 são apresentados os trabalhos relacionados. O Capítulo 4, descreve a metodologia proposta nesta pesquisa para coletar evidências observáveis da hipótese estudada. No Capítulo 5 são apresentados e discutidos os resultados obtidos e por fim no Capítulo 6 é descrito a com base nos resultados a conclusão desta investigação.

[2] http://up-for-grabs.net

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo introduzir ao leitor conceitos que foram utilizados na elaboração deste trabalho, a fim de auxiliar na compreensão dos assuntos abordados no mesmo. As definições e fundamentos descritos a seguir baseiam-se fontes literárias devidamente documentadas.

2.1 Software Livre

Software Livre (SL) é um movimento que ganhou notoriedade a partir do ano 1985, após o surgimento da *Free Software Foundation* (FSF), uma organização fundada por Richard Stallman, sem fins lucrativos, e que tem como missão promover mundialmente a liberdade de usuários de computadores, incentivando a erradicação de restrições sobre a cópia, estudo e modificação de programas de computadores (SILVA et al., 2010).

De acordo com Stallman (2002), para ser designado como SL, um software deve respeitar quatro liberdades essenciais dos usuários:

1. A liberdade de executar o programa como desejar.
2. A liberdade de estudar o código-fonte do programa e realizar alterações.
3. A liberdade de distribuir cópias do programa para outras pessoas.
4. A liberdade de distribuir cópias de suas versões modificadas.

A FSF reitera a definição de Stallman (2002) e define SL como: "Aquele que vem com permissão para qualquer um copiar, usar e distribuir, com ou sem modificações, gratuitamente ou por um preço. Em particular, isso significa que o código-fonte deve estar disponível."

Ambas as definições foram elaboradas de forma que toda a comunidade pudesse beneficiar-se e contribuir com projetos de SL. Com a popularização do conceito software livre, originaram-se muitos projetos que exercem tais definições, assim como alguns projetos antes proprietários que passaram a adotá-las, disponibilizando e permitindo a utilização de seus códigos fontes para a comunidade, passando assim a serem considerados projetos de SL. Aragon[1], Auctus[2], Box[3] e Java *Enterprise Edition*[4] são alguns exemplos de projetos que passaram por essa transição.

Outra organização importante para a comunidade de SL é a *Open Source Initiative* (OSI), fundada em 1998 como uma organização geral de educação e advocacia a fim de representar a comunidade de SL e instigar uma aproximação de entidades comerciais com o SL. Atualmente esta organização atua com foco em estabelecer de forma consciente as definições de SL mundialmente. Sua principal atividade é certificar quais licenças realmente se enquadram como licenças de SL, assim como promover a divulgação do SL e suas vantagens tecnológicas e econômicas (OSI, 1999).

[1] https://blog.aragon.one/announcing-our-migration-to-an-open-source-messaging-platform-420b25e74284
[2] https://blog.auctus.org/announcing-our-migration-to-an-open-source-messaging-platform-d061844837d5
[3] https://blog.box.com/blog/announcing-box-open-source
[4] https://blogs.oracle.com/theaquarium/opening-up-ee-update

Um dos projetos mais populares e bem sucedidos de SL é o GNU/Linux, um Sistema Operacional (SO) multiusuário e multitarefas, baseado no Unix[5], compatível com processadores de 32 e 64 bits e concebido principalmente pela união dos esforços de Linus Benedict Torvalds e Richard Matthew Stallman que almejavam criar um SO que correspondesse as definições de SL. Com o passar dos anos, somado ao trabalho árduo e voluntário de centenas de contribuidores, esse objetivo foi alcançado, dando origem a um SO constituído pelo núcleo Linux e um extenso conjunto de ferramentas de SL desenvolvidas no projeto GNU *Project*[6].

O código-fonte do núcleo deste SO (Linux) encontra-se disponível em <https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/> e graças a contribuições voluntárias está constantemente evoluindo. Atualmente este SO representa uma parcela de aproximadamente 3% do mercado mundial de SOs, sendo está uma fração bem significativa, considerando que o restante deste mercado é constituído predominantemente por produtos de organizações privadas como a Microsoft (Windows) e a Apple (Mac OS) (NETMARKETSHARE, 2021).

Os benefícios dessa modalidade não são restritos apenas a comunidade de SL. Como as definições não fazem referência a custos envolvidos, é possível que um SL não seja gratuito. Como Ferraz (2002) defende, um SL também pode ser uma fonte de vantagem estratégica para o setor privado, trazendo benefícios práticos, independentemente de posições filosóficas ou princípios morais. Algumas das vantagens apontadas são: software livre oferece maior segurança, possibilita a inovação e estimula a adoção de soluções baseadas em padrões abertos (FERRAZ, 2002).

2.2 GitHub

Mesmo grandes projetos de SL como, Ruby[7], MongoDB[8], e Jenkins[9], dependem de contribuições voluntárias de desenvolvedores para se manterem atualizados e continuarem evoluindo. Boa parte desses projetos disponibilizam seu código-fonte em serviços de hospedagem de código como o GitHub, que é uma plataforma que surgiu justamente com o intuito de fornecer suporte à atividade de desenvolvimento de SL. Além da hospedagem, esta plataforma oferece serviços de controle de versão, rastreamento de tarefas e interação social entre os contribuidores. O GitHub também é muito utilizado por empresas privadas e desenvolvedores para hospedarem seus projetos.

No GitHub contribuidores são capazes de interagir com projetos de SL, podendo propor mudanças no código acondicionado no repositório principal do projeto sem necessariamente alterá-lo ou terem permissão de gerenciamento sobre o mesmo. Isto é possível graças ao sistema de gerenciamento de código-fonte conhecido como Git, cujo a documentação encontra-se disponível em: <https://git-scm.com/doc>. Entender o funcionamento do Git é importante para que os usuários do GitHub sejam capazes de usufruir mais eficientemente de suas funções.

[5] http://www.unix.org/what_is_unix.html
[6] https://fsfe.org/freesoftware/basics/gnuproject.pt.html
[7] https://github.com/ruby/ruby
[8] https://github.com/mongodb/mongo
[9] https://github.com/jenkinsci/jenkins

Os comandos: *fork*, *branch*, *checkout*, *clone*, *pull*, *commit*, *push* e *merge* explicados sucintamente na Tabela 2.1, são alguns dos principais comandos do sistema Git utilizados no processo de uma contribuição.

Comando	Descrição
fork	Cria no perfil pessoal do usuário uma cópia referente ao repositório ao qual foi executado. É similar a um *git clone* porém a cópia é completamente independente.
git branch	Gera uma cópia, associada porém independente, da versão do repositório no exato momento em que o comando foi executado.
git checkout	Este comando está relacionado ao comando *branch* e é utilizado para transitar entre os ramos disponíveis em um repositório.
git clone	Cria no diretório onde foi executado, uma cópia do repositório designado.
git pull	Atualiza a versão do repositório armazenada no dispositivo do usuário, com as atualizações efetuadas na versão hospedada no GitHub.
git commit	Este comando precede o *push* e serve para gravar mudanças realizadas na versão local do repositório.
git push	Este comando atualiza a versão do repositório hospedada no GitHub com o conteúdo atualizado na versão armazenada localmente.
git merge	Concatena dois ou mais ramos de um repositório.

Tabela 2.1. Breve descrição dos principais comandos do Git e GitHub.

Fonte: https://git-scm.com/doc.

O GitHub é um dos principais responsáveis pela popularização do modelo de desenvolvimento baseado em *pull request.* Atualmente este modelo é massivamente adotado por projetos de SL hospedados em ambientes de codificação colaborativos como o GitHub. Neste modelo, projetos recebem contribuições de colaboradores externos por meio de *pull requests*[10] (YU et al., 2016; GOUSIOS et al., 2014).

No GitHub o mecanismo de *pull requests* também permite que usuários discutam sobre propostas de contribuições submetidas. O GitHub oferece múltiplas funções de interação social, onde contribuidores podem, acompanhar o desenvolvimento de tarefas dos projetos de seu interesse, seguir o perfil de outros desenvolvedores e, até mesmo, comentar contribuições alheias.

Outro importante serviço disponível no GitHub é o rastreamento de tarefas (*issue tracker*), que pode ser utilizado como forma de interação entre os mantenedores de um projeto e os contribuidores externos. As tarefas podem ser gerenciadas em uma das abas do painel do repositório de um projeto onde é possível: visualizar as tarefas em forma de lista, criar e aplicar rótulos para tarefas a serem categorizadas ou atribuídas a usuários, pesquisar, ordenar e filtrar tarefas assim como encerrá-las por meio de mensagens de *commit*

Os contribuidores podem criar tarefas para diversos fins, como registrar sugestões ou relatar problemas identificados no projeto, assim como os mantenedores podem utilizá-las para documentar

[10] *Pull request* é o termo utilizado para denominar a ação de propor a integração de mudanças a um repositório.

assuntos pendentes ou publicar para a comunidade requisições de problemas específicos dos quais se aguarda contribuição, o que aumenta a eficiência da organização promovendo a evolução do projeto como um todo (CABOT et al., 2015).

As tarefas ficam registradas juntamente ao repositório e possuem diversos atributos como: título, descrição, assinantes e dentre outros o rótulo. Uma tarefa pode possuir mais de um rótulo, que basicamente é uma expressão anexada a tarefa. *Pull-requests* também podem possuir rótulos atribuídos.

Um rótulo pode ter diversas funcionalidades, como: definição de prioridades das tarefas, especificar a complexidade de uma tarefa, informar o escopo da tarefa, iniciar discussões sobre assuntos pertinentes ao projeto e até mesmo recomendar tarefas para um perfil específico de contribuidores.

Essa diversidade de recursos fez com que plataformas como o GitHub mudassem o modo ao qual desenvolvedores de software se comunicam, colaboram e contribuem com projetos de código aberto (DIAS et al., 2016). Em seu trabalho, Dias et al. (2016) investigam se projetos de SL que migraram seus repositórios de repositórios privados para ambientes de codificação colaborativos como o GitHub tiveram um aumento significativo em relação a atratividade de novos contribuidores.

2.3 Contribuições em Projetos de Software Livre

Cada vez mais desenvolvedores ingressam à comunidade de SL. Estes compartilham do princípio ético de que conhecimento deve ser compartilhado. A ideia de contribuir para um projeto, muitas vezes sem uma recompensa direta associada, pode parecer um pouco contraintuitiva. Surpreendentemente isto não inibe desenvolvedores do mundo todo de contribuírem para projetos de SL. Essas contribuições podem ser motivadas por diversos fatores (PINTO et al., 2016; GHOSH, 1998; HANN et al., 2002; GHOSH, 2005).

Na maior parte das profissões o melhor jeito de aprender e aperfeiçoar suas habilidades é a prática. Contribuir para projetos de SL pode oferecer diversos benefícios para desenvolvedores, pois provém não só a oportunidade de aprimorarem sua proficiência em programação, interpretar código alheio, encontrar e corrigir erros, mas também de aprender com os comentários recebidos de desenvolvedores mais experientes (BITZER et al., 2007; LERNER; TIROLE, 2002).

Geralmente estas são atividades que desenvolvedores vivenciam quando trabalham em empresas de tecnologia. Portanto, podem ser experiências úteis e valiosas, podendo até mesmo serem requisitadas por empresas para uma vaga de emprego, tanto de estágios quanto para cargos efetivos.

Um fluxo contínuo de contribuidores e seu engajamento em atividades de desenvolvimento são fatores cruciais para o sucesso dos projetos de SL (KROGH et al., 2003). Deste modo, a comunidade de SL busca constantemente por abordagens que auxiliem desenvolvedores a realizarem contribuições nestes projetos.

Todavia, apesar da existência de várias destas abordagens, novatos ainda enfrentam uma diversidade de dificuldades ao tentarem submeter uma contribuição a um projeto de SL. Sendo assim,

um grande desafio para a comunidade é prover maneiras de dar suporte ao ingresso de novatos (STEINMACHER et al., 2014).

Estas dificuldades não só atrapalham o processo de contribuição dos novatos, como prejudicam o processo de desenvolvimento do projeto, dado que um contribuidor mais experiente poderia estar solucionando problemas mais complexos do projeto com o tempo desperdiçado na solução de tarefas triviais que poderiam ser solucionadas por novatos.

O processo envolvido na submissão de uma contribuição para um projeto não é trivial, esta é uma das dificuldades enfrentadas por contribuidores apontadas no trabalho de Steinmacher et al. (2014). Além disso, nem sempre o processo de contribuição culmina na integração da solução proposta pelo contribuidor ao repositório, uma vez que a solução proposta pode não satisfazer os proprietários do projeto (GOUSIOS et al., 2014).

Também é importante saber distinguir o escopo destes comandos para saber qual é mais indicado para determinadas situações, por exemplo, quando executado, o comando *fork*, gera no acervo de repositórios do usuário, uma cópia do repositório ao qual foi aplicado.

O *fork* pode ser executado por qualquer usuário em qualquer repositório aberto, já o comando *branch* só pode ser executado pelos proprietários de um repositório, e gera um ramo do repositório onde foi executado, que basicamente é uma cópia da versão em que código se encontrava no exato momento em que o comando foi executado, geralmente é utilizado para que o proprietário possa desenvolver funções paralelamente ou realizar testes. Sendo assim o *fork* normalmente é utilizado em contribuições submetidas por usuários que não são membros de um projeto e o *branch* exclusivamente por membros do projeto.

A Figura 2.1 apresenta um fluxograma do processo típico executado para a realização de uma contribuição em repositórios de código aberto hospedados no GitHub.

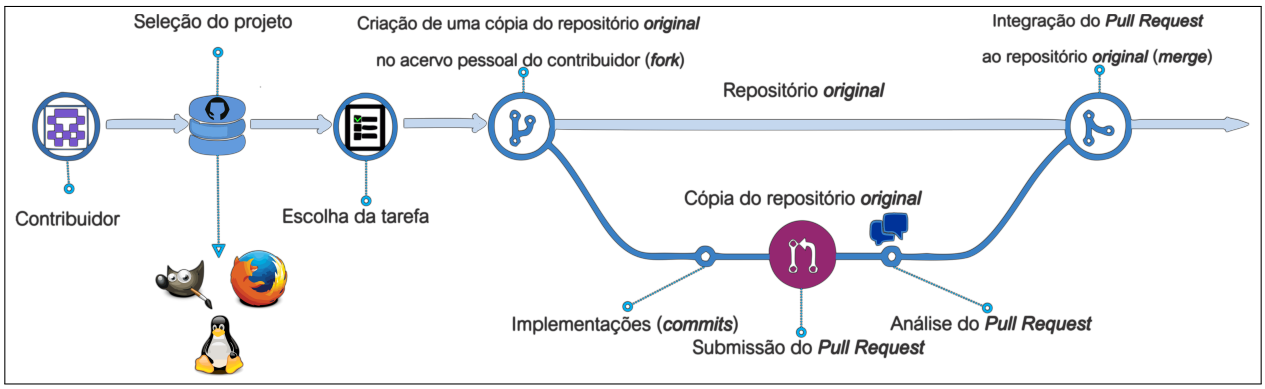

Figura 2.1. Processo de contribuição no ambiente GitHub

Fonte: Autoria Própria - Baseado em https://guides.github.com/introduction/flow.

O processo descrito na imagem procede da seguinte maneira. Primeiramente o contribuidor seleciona o projeto de SL o qual deseja contribuir. Esta escolha pode ser influenciada por diversos fatores, como: afinidade com as tecnologias utilizadas no projeto, popularidade do projeto, eventos

que premiam contribuidores, entre outros (HERTEL et al., 2003). Entender os fatores envolvidos nessa seleção é importante para que os projetos possam se aprimorar a fim de atrair mais contribuidores.

Posteriormente à seleção do projeto, é necessário definir qual tarefa o contribuidor pretende solucionar. A investigação proposta por este trabalho situa-se nesta etapa. Encontrar uma tarefa adequada pode ser a chave para que o contribuidor obtenha sucesso ao fim do processo (STEINMACHER et al., 2015). As tarefas ficam registradas na aba de tarefas do repositório e podem relatar desde erros no código-fonte até funcionalidades que precisam ser implementadas.

Para implementar e submeter a solução da tarefa escolhida é preciso preparar o ambiente de desenvolvimento bem como entender as funções e como utilizar os comandos do sistema Git. Esta etapa pode proceder de duas maneiras. Se o contribuidor for membro do projeto ele pode criar um ramo secundário do repositório por meio do comando *branch* e implementar suas mudanças nesse ramo para assim realizar o *pull request*. A outra maneira é para contribuidores externos, a única diferença é que, em vez de executar o *branch* no repositório, realiza-se um *fork*, isto é, cria-se uma cópia deste repositório no acervo pessoal de repositórios do contribuidor.

As implementações realizadas no ramo secundário ou na cópia gerada pelo *fork* no repositório pessoal do contribuidor não afetam o ramo ou o repositório principal do projeto e vice-versa, o que proporciona aos proprietários do projeto mais segurança em relação a integridade do repositório principal e concede ao contribuidor a liberdade de experimentar quaisquer mudanças desejadas.

Após o contribuidor implementar em seu repositório uma solução para a tarefa escolhida, este precisa realizar um *pull request*, para que se aceito, o código proposto possa ser integrado ao repositório principal (*merge*). Antes do *merge*, é comum que cada *pull request* submetido seja analisado pelos membros do projeto. Nessa etapa, o contribuidor pode receber sugestões para melhorar a solução proposta e porventura submetê-la novamente.

Se a requisição for aprovada o ramo secundário é integrado ao ramo principal, ou, no caso de contribuidores externos, apenas as alterações realizadas na versão hospedada no repositório do contribuidor são integradas ao repositório principal do projeto (*merge*). A integração do código, assim como os problemas técnicos que podem vir a ocorrer, são resolvidos manualmente por quem aprovou o *pull request*. Assim concluísse a jornada pela qual um contribuidor se submete a fim de colaborar com projetos de SL.

Pode-se distinguir os perfis de contribuidores existentes pela frequência de contribuição que estes realizam. Para o escopo deste trabalho define-se como novato todo aquele que submeteu um *pull request* para um projeto pela primeira vez, sendo este aceito ou não e independente de sua competência como desenvolvedor ou seu histórico de contribuição em outros projetos. Novatos enfrentam diversas dificuldades e obstáculos quando iniciam o processo descrito na Figura 2.1, o que resulta em uma baixa taxa de retenção de contribuidores em projetos de SL (STEINMACHER et al., 2013).

A partir do momento em que este contribuidor, antes novato, passa a contribuir com maior frequência a um determinado projeto, este pode ser classificado em dois outros perfis de contribuidores: casuais e assíduos. Os contribuidores casuais submetem contribuições com pouca

frequência, porém esporadicamente, já os assíduos, que normalmente são mantenedores do projeto, realizam contribuições constantemente. Estes perfis são tão importantes para a comunidade quanto os novatos. De acordo com (GOUSIOS et al., 2014), cerca de 7% dos *pull requests* realizados no GitHub originaram-se de contribuições casuais.

2.4 Programas de Incentivo à Novatos

A comunidade de SL promove constantemente programas que dão suporte a contribuidores com o intuito de atraí-los e preferencialmente retê-los. Algum desses programas envolvem incentivos financeiros ou outros modos de recompensa.

Google Summer of Code

Google Summer of Code (GSoC) é um programa global que tem como objetivo promover a aproximação de estudantes com o desenvolvimento de SL. A ideia consiste em estimular os estudantes a trabalharem com projetos de SL durante os 3 meses de férias da faculdade. O *Google* oferece uma premiação em dinheiro para os estudantes que obterem sucesso ao fim de todas as etapas do GSoC.

Na primeira etapa do evento, o estudante basicamente precisa estabelecer um vínculo com uma organização mentora sendo preciso escolher e submeter uma proposta para uma das organizações de SL cadastradas no evento. Depois de revisarem e selecionarem as propostas realizadas, as organizações se unem aos estudantes selecionados para estabelecer um projeto e suas metas. No primeiro mês o propósito é que os estudantes aprendam sobre a comunidade de sua organização mentora.

Após isso os estudantes iniciam o desenvolvimento de seus projetos para o GSoC. Ao fim do desenvolvimento, o estudante e a organização submetem avaliações uns dos outros. O estudante submete o seu código e avaliação final dos mentores. A organização mentora revisa o código submetido pelo estudante e determina se este obteve sucesso em completar seu projeto GSoC, notificando-o do resultado ao fim do evento.

Hacktoberfest

Anualmente, no mês de outubro, a *Digital Ocean*, empresa que provê infraestrutura para computação em nuvem, formaliza a sua parceria com o GitHub para realização do *Hacktoberfest*. Este evento dura o mês inteiro de outubro e busca incentivar a participação em projetos de SL. Qualquer pessoa que queira pode participar da sua própria casa. As regras do *Hacktoberfest* são bem simples e menos burocráticas que o GSoC.

Para participar do *Hacktoberfest* um contribuidor deve se cadastrar e abrir quatro *pull requests* em qualquer repositório público no GitHub, independente se esse projeto tem ou não tarefas marcadas com o rótulo do evento. Participantes que realizam *pull requests* fraudulentos são desclassificados. Os participantes não precisam necessariamente terem seus *pull requests* aceitos para que estes sejam

contabilizados. Ao fim do evento os participantes que atingirem o objetivo são premiados com uma camiseta.

2.5 Iniciativas de Suporte à Novatos Baseadas em Anotação de Tarefas

Dos recursos do GitHub descritos na Seção 2.2, o mais relevante para o contexto desta pesquisa é o serviço que permite a aplicação de rótulos em tarefas. A utilização de rótulos torna mais eficiente o gerenciamento das tarefas de um projeto (CABOT et al., 2015). Alguns projetos como o Ansible[11], Flasky[12] e o PowerShell[13], utilizam esse recurso disponível no rastreio de tarefas para designar uma tarefa como recomendada para novatos.

A Figura 2.2 demonstra a utilização do recurso de filtro de tarefas aplicado a uma das variações de rótulos de recomendação para novatos conhecida, que retorna ao usuário uma lista das tarefas anotadas com o rótulo designado.

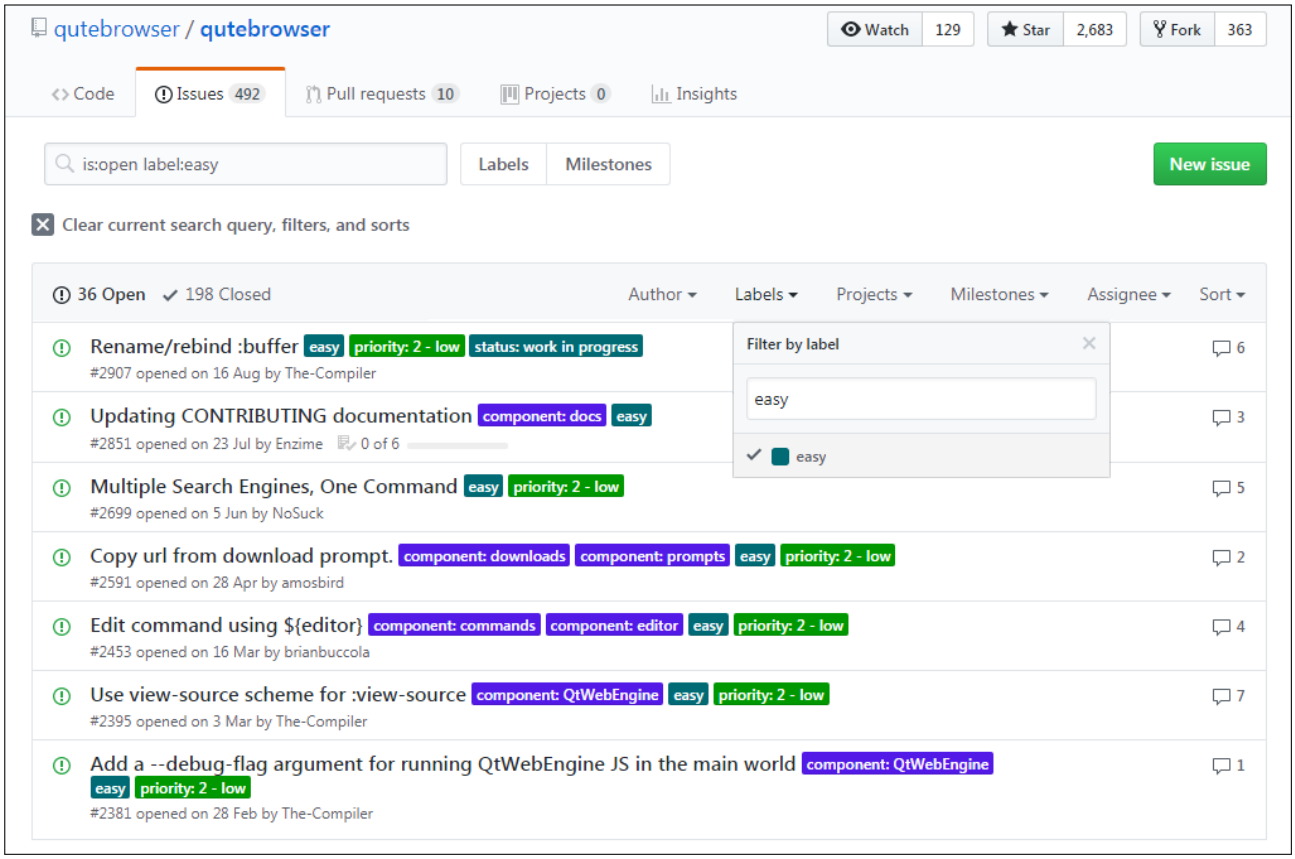

Figura 2.2. Filtragem de tarefas pelo rótulo "*easy*"

Fonte: https://github.com/qutebrowser/qutebrowser/issues.

[11] https://github.com/ansible/ansible/issues?q=is:open+is:issue+label:easyfix
[12] https://github.com/pallets/flask/issues?q=is:open+is:issue+label:"good+first+issue"
[13] https://github.com/PowerShell/PowerShell/issues?q=is:open+is:issue+label:Up-For-Grabs

É interessante destacar que as tarefas podem ser anotadas com mais de um rótulo. No exemplo apresentado na Figura 2.2 o rótulo “priority: 2 - low” é usado para determinar a prioridade da resolução de uma tarefa como baixa e os rótulos que iniciam com a expressão “component:” especificam o escopo técnico da tarefa. Deste modo, os rótulos que compõem uma tarefa já rotulada como recomendada para novatos podem estar relacionados à características que podem ser usadas para reconhecer outras tarefas indicadas para novatos que ainda não foram rotuladas.

Sabendo que o sistema de rótulos vêm sendo utilizado em alguns repositórios para indicar tarefas adequadas para novatos, vários projetos, desenvolvidos pela comunidade, começaram a listar e divulgar estes repositórios ou diretamente tarefas já rotuladas. Estas iniciativas tem como objetivo auxiliar novatos a encontrarem uma tarefa adequada para primeira contribuição, promovendo assim a receptividade em projetos de SL.

Up-For-Grabs

O *Up-For-Grabs* é uma das iniciativas que listam projetos de SL que praticam a rotulação de tarefas indicadas para novatos em seus repositórios hospedados no GitHub. Essa iniciativa visa atrair novos contribuidores para os projetos de SL, partindo do pressuposto de que a utilização de rótulos para novatos é uma prática com potencial para atingir tal objetivo.

A interface do *Up-For-Grabs* apresenta, para cada um dos projetos cadastrados, as seguintes informações: nome, breve descrição do projeto, rótulos que o associam a características como linguagem ou área de atuação, rótulo oficial utilizado para designar uma tarefa indicada para novatos e a quantidade de tarefas para novatos pendentes em seu repositório.

Como exibido na Figura 2.3, o *Up-For-Grabs* também oferece dois sistemas de filtragem dos projetos cadastrados: um por nome e outro pelos rótulos. Projetos que desejam ser listados devem realizar um *pull request* ao repositório do *Up-For-Grabs*[14] com as informações de cadastro do projeto.

[14] https://github.com/up-for-grabs/up-for-grabs.net

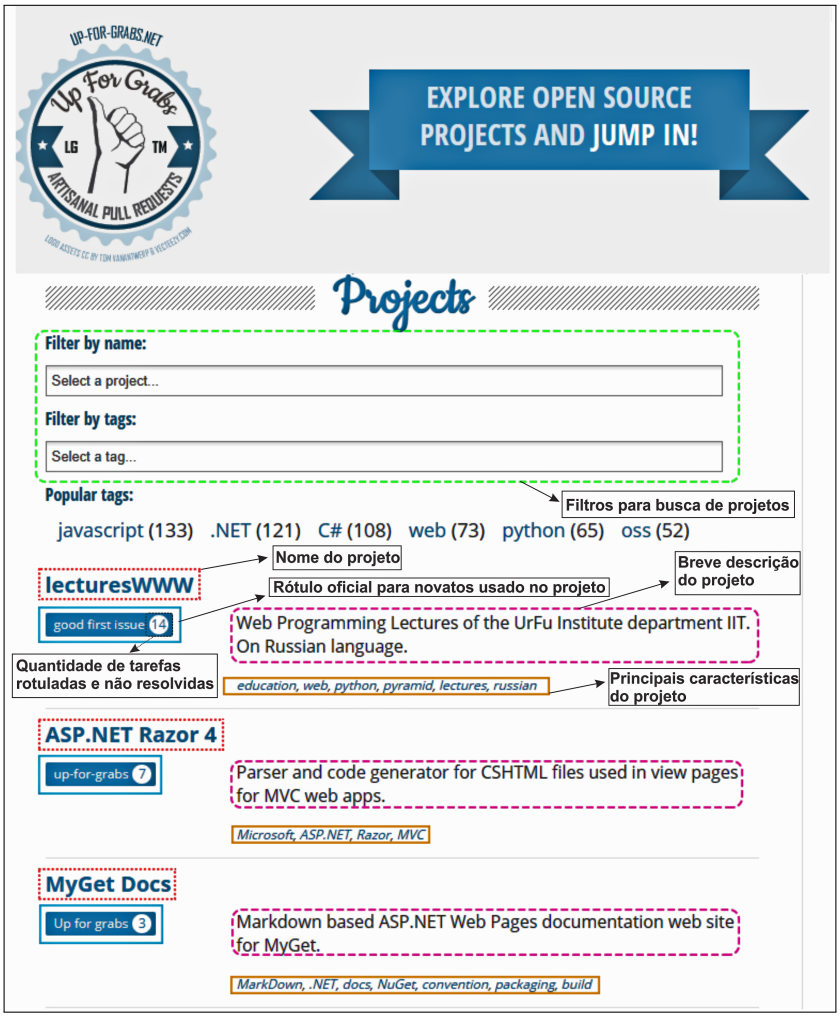

Figura 2.3. Interface da plataforma *Up-For-Grabs*

Fonte: http://up-for-grabs.net.

Awesome-For-Beginners

O *Awesome-For-Beginners* é mais um exemplo de iniciativa que divulga repositórios que utilizam rótulos para novatos. Porém, diferente do *Up-For-Grabs*, a lista de repositórios adotantes desta prática é disponibilizada por meio de um repositório hospedado no GitHub, disponível em: <https://github.com/MunGell/awesome-for-beginners>, sendo esta, mais uma das opções criadas pela comunidade para dar suporte aos novatos que buscam uma tarefa adequada para contribuir.

First Timers Only (Bot)

O Twitter é uma das redes sociais mais populares da atualidade. nela os usuários podem enviar e receber atualizações pessoais de outros contatos, em textos de até 280 caracteres, conhecidos como "*tweets*". Membros da comunidade de SL desenvolveram *bots*[15] para o Twitter que realizam a divulgação automática, por meio de *tweets*, de tarefas anotadas com o rótulo *"first-timers-only"* no

[15] É uma aplicação de software concebida para simular ações humanas repetidas vezes de maneira padrão, da mesma forma como faria um robô.

GitHub. Toda vez que algum usuário do GitHub cria uma nova tarefa e atribui a ela este rótulo, o perfil $first\_tmrs\_only$[16] publica um *tweet* divulgando-a. A Figura 2.4 apresenta uma captura de tela de um dos *tweets* de divulgação realizados por este perfil.

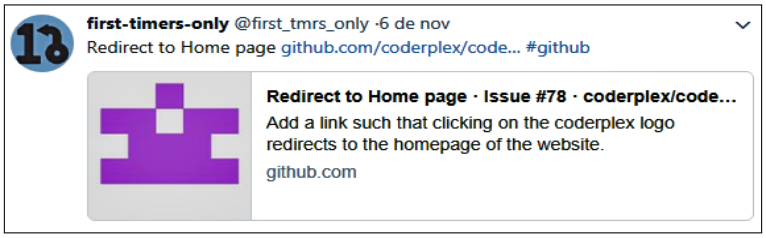

Figura 2.4. Exemplo de divulgação de tarefas indicadas para novatos no Twitter

Fonte: https://twitter.com/first_tmrs_only/status/927575569453154314.

Busca Manual de Tarefas Recomendadas para Novatos

No campo de pesquisa disponível em qualquer página do GitHub, os usuários podem buscar conteúdos específicos entre os milhares de repositórios hospedados no GitHub. Dentre outras possibilidades, como buscar por usuários, repositórios ou até mesmo *commits* específicos, esse sistema viabiliza, por meio do uso de expressões de consulta, que o usuário pesquise por tarefas anotadas com rótulos para novatos, pendentes ou não. A *Uniform Resource Locator* (URL) a seguir é um exemplo de como realizar uma busca por tarefas para novatos anotadas e pendentes, com o rótulo "first-timers-only" : https://github.com/search?q=label:first-timers-only+is:issue+is:open. Evidentemente, para buscar tarefas indicadas para novatos utilizando o campo de pesquisas do GitHub, o usuário necessita conhecer previamente ao menos um dos diversos termos utilizados em rótulos para novatos.

2.6 Considerações Finais

Os assuntos abordados neste tópico servem de embasamento teórico ao leitor. Nas Seções 2.1, 2.2 e 2.3 fora apresentado uma concepção geral, da fundamentação do movimento de SL, dos recursos e do impacto de ambientes de codificação colaborativos em projetos de SL e de como funciona o modelo de desenvolvimento baseado em *pull requests*. Já as Seções 2.4 e 2.5 apresentam exemplos de projetos que buscam dar suporte a novatos em suas primeiras contribuições, o que por sua vez evidencia o quanto os membros da comunidade de SL estão se preocupando em auxiliar esse perfil de contribuidor.

[16] https://twitter.com/first_tmrs_only

3 TRABALHOS RELACIONADOS

Neste capítulo serão apresentados e discutidos sucintamente alguns dos trabalhos que abordam assuntos relacionados ao contexto desta pesquisa, de modo a prover ao leitor uma melhor compreensão do cenário ao qual essa pesquisa se situa e suas motivações.

Nos últimos anos, muitos pesquisadores se dedicaram a buscar entender quais são as razões que motivam contribuições para projetos de Software Livre (SL), assim como os motivos que podem vir a desestimular um individuo a se tornar um contribuidor. Entendendo essas motivações é possível elaborar estratégias para aumentar a atratividade desses projetos.

Em sua pesquisa, por meio de análise qualitativa aplicada a entrevistas conduzidas com contribuidores de SL novatos, Steinmacher et al. (2015) identificou que a escolha de uma tarefa adequada para primeira contribuição é uma das principais barreiras enfrentadas por novatos que tentam contribuir para projetos de SL. Nesta pesquisa foi evidenciado que a maioria dos novatos não se sentem confiantes o suficiente para escolher uma tarefa para contribuir pela primeira vez. Os novatos relataram que precisam de informações detalhadas sobre as tarefas ou preferencialmente orientação de membros da comunidade do projeto para escolher uma tarefa adequada.

Neste contexto, várias estratégias vem sendo exploradas e tentadas pela comunidade buscando tornar os projetos mais amigáveis e atrativo para novatos. O GitHub por exemplo sugere aos projetos que mantenham os arquivos *README.md* e *CONTRIBUTING.md* em seus repositórios para os tornarem mais receptivos para novatos (DIAS, 2017), esses arquivos normalmente contém informações gerais do projeto além de instruções de como executá-lo e como realizar uma contribuição.

Outra estratégia sugerida pelo GitHub é utilização de rótulos como o *good first issue* para marcar tarefas adequadas para novatos. Cada vez mais projetos vem adotando essa prática nos últimos anos, Alderliesten e Zaidman (2021) estima que 43.8% dos projetos hospedados no GitHub já adotam esta prática. Este mesmo trabalho explorou essa estratégia e apontou alguns possíveis problemas em sua execução. Alderliesten e Zaidman (2021) descobriram que em projetos que utilizam a prática, a quantidade média de tarefas com rótulos de indicação para novatos é de apenas 1.5% e que das tarefas rotuladas e consideradas como resolvidas, apenas 32.5% foram resolvidas realmente por novatos. Também foi verificado que o tipo das tarefas indicadas não corresponde satisfatoriamente as tarefas que de fato são resolvida pelos novatos. Desde modo concluíram que apesar da prática colaborar na integração do novato, mudanças podem ser feitas para melhorar a indicação das tarefas.

Além do tempo e esforço necessário para identificar e rotular uma tarefa como indicada para novato, o critério utilizado para julga-lá como "fácil" pode variar de acordo com o julgamento do responsável pela atribuição do rótulo na tarefa. Em seu trabalho Huang et al. (2021) propõem uma caracterização das tarefas que são indicadas para novatos de modo que por meio de técnicas de aprendizado de máquina, seja possível executar essa atividade de forma automática. Neste trabalho análises quantitativas evidenciaram que tarefas indicadas para novatos possuem diferença significativa

das demais quando consideradas características relacionadas a semântica, legibilidade e riqueza de informações na descrição da tarefa. Deste modo foi possível desenvolver um modelo de predição que baseando-se nessas características foi capaz de prever eficientemente outras tarefas que poderiam ser indicadas para novatos. Este tipo de ferramenta pode aumentar significativamente a eficiência dessa estratégia, visto que exigiria menos esforço dos projetos para sua aplicação e provavelmente aumentaria a quantidade de tarefas recomendadas.

Tan et al. (2020) também conduziram uma investigação sobre esta prática. Neste estudo foram analisados 816 projetos do GitHub e suas tarefas. Assim como as pesquisas apresentadas anteriormente nesta seção, foi identificado que esta prática vem sendo cada vez mais adotada e que existem alguns problemas relacionados a quantidade insuficiente de tarefas rotuladas e a classificação inapropriada dessas tarefas, isto é, tarefas que foram indicadas mas que os novatos não consideram uma boa tarefa para primeira contribuição.

Diferente de Huang et al. (2021), neste trabalho concluiu-se que não existe diferença significativa entre as tarefas indicadas para novatos e as demais. Isso demonstra que apesar de existirem conclusões em comum entre as pesquisas, como é o caso da constatação da crescente adesão da prática e de alguns problemas relacionados a baixa presença de tarefas indicadas para novatos, não existe um consenso em relação a hipótese de que é possível mapear características para distinguir essas tarefas, e portanto mais estudos devem ser conduzidos neste sentido.

De modo geral os trabalhos abordados nessa seção buscaram entender as dificuldades, necessidades e motivações dos novatos em comunidades de SL e também exploraram estratégias que propõem minimizar essas dificuldades, como a utilização de indicação de tarefas para novatos. Diferente dos demais trabalhos cujo foco maior é na identificação de características das tarefas indicadas para novato, este trabalho propõe uma investigação mais direcionada para validação do impacto da prática na atratividade dos projetos de SL.

4 METODOLOGIA

Este capítulo descreve a metodologia executada a fim de investigar a validade da hipótese de que, a rotulação de tarefas "indicadas para a primeira contribuição" pode aumentar o engajamento de novatos em projetos de Software Livre (SL) que a empregam em seus repositórios. A Seção 4.1 descreve o objetivo e a motivação deste trabalho. Na Seção 4.3 são descritas as questões elaboradas para conduzir essa pesquisa, seguido pela Seção 4.4 que apresenta o processo de seleção dos repositórios que foram avaliados. Já a Seção 4.5 descreve a maneira como foi realizada a coleta dos dados.

O diagrama disposto na Figura 4.1 exibe uma visão geral do fluxo de execução das etapas que são apresentadas nessa seção.

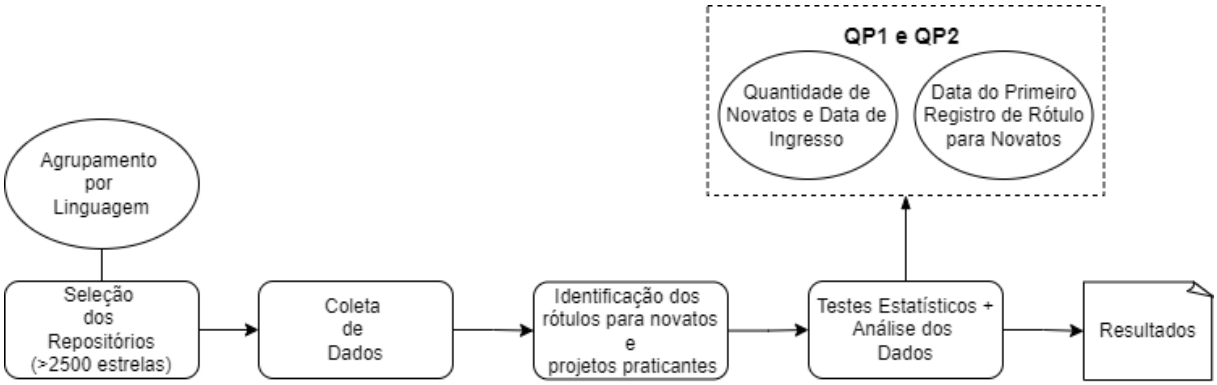

Figura 4.1. Diagrama das etapas da metodologia proposta

Fonte: Autoria Própria.

4.1 Objetivo

O objetivo deste trabalho é verificar a influência que a utilização de rótulos para indicar tarefas a novatos exerce na atratividade de repositórios de código aberto. Supõe-se que esta prática tem impacto positivo na atratividade de novatos dos projetos, visto que os rótulos facilitam a seleção de uma tarefa para o novato contribuir. Porém, pouco se sabe sobre o real impacto desta prática, deste modo, essa pesquisa propõe uma investigação estatística sobre o mesmo a fim de entendê-lo melhor.

Esta investigação foi conduzida por meio de uma análise comparativa, na qual foram aplicados os testes de hipóteses estatísticos *Mann-Whitney-Wilcoxon* (MWW) e *Delta de Cliff* em distribuições relativas ao ingresso de novatos no período anterior e posterior a adesão da prática. Também foram comparados repositórios que aderiram a prática com repositórios que não o fizeram. O elemento em comum de ambas as avaliações é a separação dos dados pelo critério de utilização ou não da prática de forma que, com os resultados dos testes estatísticos fomos capazes inferir, primeiramente, se existe diferença estatística na comparação entre as distribuições, e caso sim, o tamanho desta diferença.

4.2 Identificação dos Rótulos para Novatos e dos Projetos Praticantes

Observando os repositórios catalogados no *Up-For-Grabs*, notou-se uma variação muito grandes dos termos utilizados como rótulos para designar que uma determinada tarefa é recomendada para novatos.

O *Up-For-Grabs* registra qual é o rótulo oficial utilizado no repositório, tornando mais simples a busca do contribuidor. Porém esta variação de termos pode vir a ser um grande problema para os novatos que não estão cientes da existência de iniciativas que listam repositórios que adotam a prática ou que estão em busca de tarefas recomendadas para novatos em algum repositório que não se encontra listado por iniciativas como o *Up-For-Grabs*. Este problema é decorrente da ausência de um padrão que determine qual termo deve ser utilizado nos rótulos para novatos em todos os projetos que utilizam esse recurso. Além disso, existem projetos que não especificam em sua documentação qual termo utilizam a este fim.

A ausência de uma convenção que determine um termo padrão a ser utilizado na rotulação de tarefas recomendadas para novatos pode ter como consequência a invalidação do objetivo almejado pela prática, que é auxiliar novatos a encontrarem uma tarefa adequada para contribuírem. É possível que um desenvolvedor que deseja iniciar uma primeira contribuição ao projeto não seja capaz de identificar o rótulo de tarefas para novatos específico do repositório e acabe não realizando a contribuição.

Além disso, em alguns casos também percebemos a utilização de múltiplos rótulos para novatos em um repositório. Como exemplo, a Figura 4.2 apresenta um recorte do conjunto de rótulos do projeto Certbot[1], onde pode-se verificar a existência de 3 rótulos com termos normalmente associados a rótulos para novatos.

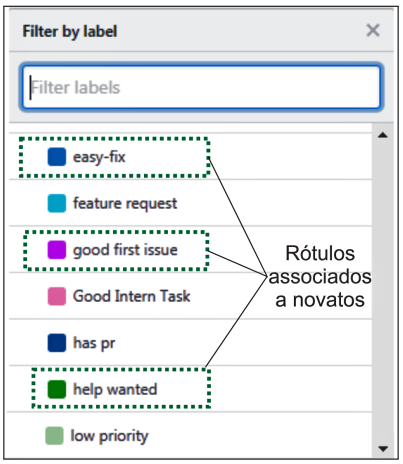

Figura 4.2. Exemplo da ocorrência de múltiplos rótulos para novatos em um repositório
Fonte: Repositório do Projeto Certbot[1].

[1] https://github.com/certbot/certbot/labels

O GitHub parece estar ciente tanto da aplicação de rótulos para recomendação de tarefas para novatos quanto do problema de variação de termos entre estes rótulos, pois recentemente passou a sugerir os termos "*help wanted*" e "*good first issue*" para serem utilizados em tarefas recomendadas para novatos. A Figura 4.3 expõe o anúncio de sugestão, que pode ser encontrado no GitHub, de termos a serem utilizados nos rótulos para indicar tarefas para novatos.

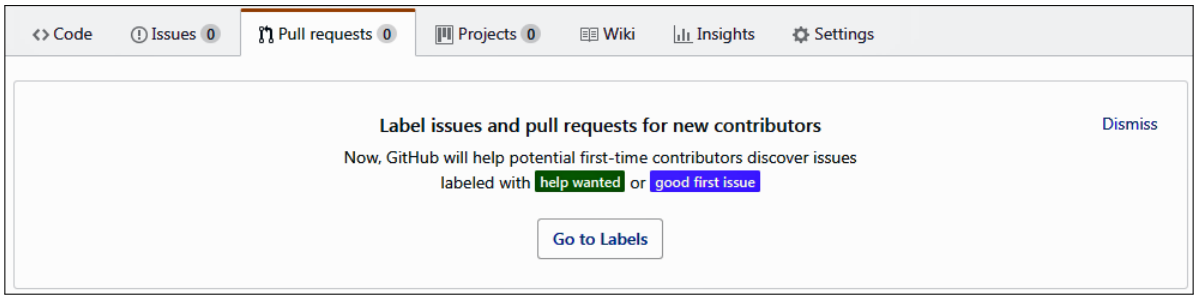

Figura 4.3. Sugestão de rótulos para novatos

Fonte: https://github.com/.

A abordagem definida para este tópico consiste no mapeamento das variações de termos utilizados nos rótulos para novatos dentro dos repositórios selecionados para este estudo, cujo processo de seleção encontra-se descrito na Seção 4.4 e também dos termos registrados no *Up-For-Grabs*.

Foram catalogados, manualmente, os termos identificados para se referir a uma tarefa para novatos nos repositório selecionados para este estudo. A identificação destes termos foi realizada de forma empírica, baseando-se na experiência adquirida pela observação das variações de termos encontradas nos elementos do conjunto de termos mapeados no *Up-For-Grabs* e outras comunidades semelhantes. Desta forma criou-se um segundo conjunto composto pelas ocorrências averiguadas, desta vez, nos repositórios avaliados neste trabalho.

Os termos verificados na etapa anterior passaram por um refinamento, no qual foram removidos o excesso de termos análogos. Por exemplo, foram verificados os termos: "*Help Wanted*", "*help-wanted*" e "*help wanted*", após o refinamento todas as variações se tornaram representadas pelo termo "*help-wanted*".

Este conjunto de termos possibilita com que possamos identificar por meio de *script* projetos que utilizam rótulos para novatos e portanto viabiliza os procedimentos abordados nas demais questões de pesquisa investigadas neste trabalho. Além disso, também foram quantificadas a ocorrência de utilização de cada rótulo nos projetos de forma a possibilitar uma compreensão de quais termos vem sendo mais utilizados pela comunidade e entender se os projetos vem seguindo a recomendação do GitHub.

Como apresentado na Figura 2.3 é possível encontrar registrado no *Up-For-Grabs*, para cada repositório, o termo oficial utilizado nos rótulos para designar tarefas recomendadas para novatos. Este acervo de termos foi registrado e utilizado para criar um conjunto de validação constituído por cada variação de termo para novatos identificados no *Up-For-Grabs*.

Concatenando os conjuntos relativos aos termos mapeados no *Up-For-Grabs* e aos repositórios selecionados para este estudo, obteve-se um total de 124 diferentes termos utilizados em rótulos para referir-se a uma tarefa recomendada para novatos.

A Tabela 4.1 apresenta uma amostra, ordenada alfabeticamente, dos termos obtidos ao fim deste processo. Os termos encontram-se dispostos exatamente como são utilizados em seus repositórios, o que explica a diferença entre os padrões de escrita observados, como uso de letras maiúsculas e espaçamento.

As células cinzas representam os termos que foram extraídos no *Up-For-Grabs*, portanto são oficialmente reconhecidos como termos para novatos. As demais células representam os termos verificados manualmente nos repositórios, sendo assim podem ser um falso positivo. Este problema é abordado com mais detalhes na Seção 5.3.

Accepting Merge Requests	easy pickings	Great First Issue	open for pull request
Accepting PRs	easy task	HELP NEEDED	please contribute
assigned to community	Easy to implement	help/beginner	please-help
audience - novice	easy-fix	help needed	Probably Easy
available	easy-pick	help-wanted	PRs plz!
backlog-easy	E-easy	hotlist: community-help	PRs welcome
beginner	effort1: small	Include Gardener	quick win
Beginner Suitable	error	intro issue	quick_to_fix
beginner-friendly	Feature Request (easy)	jump-in	Quickfix
Bite-size	first-contribution	Junior	ready to implement
bite-sized	first-timers-only	junior job	Simple Task
Bug (easy)	fit for beginners	level: easy	small
community	for-new-contributors	Level: minor	starter bug
complexity: you can do this	good contribution	Level:Starter	starter issue
complexity:easy	Good First Bug	LocalSupport	starter-task
contribution welcome	Good First Change	low-hanging-fruit	status/help-wanted
contribution-starter	good first step	Need Help	status: help wanted
contributor-friendly	Good First Task	need-code	stupid-question
contributor-needed	Good for New Contributors	needs-community-help	SUPPORT
d.FirstTimers	Good Volunteer Task	New Contributor	support_request
difficulty/low	good_for_contributors	newbie	trivial
difficulty/newcomer	good-first-bug	newbie-friendly	Up-For-Grabs
Difficulty: beginner	good-first-contribution	newcomer	you can help!
disposition/help wanted	good-first-patch	nice 1st contribution	your help needed
easy hack	good-first-pr	Novice Issues	you-take-it

Tabela 4.1. Amostra refinada da variação de termos de rótulos para novatos

Fonte: Autoria Própria.

Observando esta amostra fica evidente o problema que um novato pode enfrentar em sua busca por uma tarefa anotada com um rótulo de recomendação para novatos. Mesmo que alguns termos como "*beginner-friendly*", "*Good for New Contributors*", "*good first issue*" e "*fit for beginners*", sejam bastante explícitos em relação a sua finalidade, outros como "*Include Gardener*", "*low hanging fruit*" e "*bite-sized*", podem não ser tão intuitivos e facilmente identificáveis. O mapeamento desses termos pode ser utilizado por novatos para ampliar o alcance de sua busca por tarefas indicadas para primeira contribuição, utilizando o sistema de pesquisa do GitHub.

Por exemplo, variando o parâmetro "rótulo", com algum dos termos apresentados pela Tabela 4.1, na *Uniform Resource Locator* (URL): <https://github.com/search?q=label:<***rótulo***>+is:issue+is:open>, é possível encontrar diversas tarefas pendentes e indicadas para novatos em repositórios do GitHub.

Durante o mapeamento das variações de termos utilizadas nos rótulos para novatos, observou-se que uma grande parte dos projetos utilizava rótulos similares a "*help-wanted*" para outras finalidades que não à indicação para novatos, nesses casos a grande maioria possuía um outro rótulo para este fim. A Tabela 4.2 apresenta as variações do termo *help-wanted* que foram identificadas durante essa análise.

status:help-wanted	Help wanted	help wanted
help-wanted	disposition/help wanted	HelpWanted
Help-Wanted	state: help wanted (PR)	status/help-wanted
status: help wanted	type/help-wanted	Type: help-wanted

Tabela 4.2. Conjunto de variações de termos similares a *help-wanted*

Fonte: Autoria Própria.

Dos 1060 projetos utilizados nessa pesquisa, detectou-se a utilização de rótulos similares a *help-wanted* em 404 deles, e destes 287 apresentavam um segundo termo relacionado a indicação para novatos, o que sugere que este não é um bom termo para este fim pois pode causar confusão aos contribuidores. Em seu trabalho Tan et al. (2020) decidiram deletar rótulos relacionados ao termo *help-wanted* devido a incerteza deste estar realmente sendo utilizado para identificar tarefas indicadas para novatos. Devido a esta constatação e para melhor entendimento da prática investigada neste trabalho optou-se por executar os testes estatísticos descritos na **QP1** e **QP2** para dois diferentes cenários, sendo eles:

Cenário i Considerando os termos similares a *help-wanted*
Cenário ii Desconsiderando os termos similares a *help-wanted*

Para complementar a análise desta questão de pesquisa também foram classificados a quantidade de ocorrências verificadas de cada rótulo para novato nos projetos. A Tabela 4.3 apresenta uma classificação por ordem decrescente dos rótulos mais utilizados. Podemos verificar que os termos sugeridos pelo GitHub são mais utilizados do que os demais, indicando que a comunidade vem adotando esta sugestão. Esta verificação também foi executada agrupando os projetos por linguagens,

e para cada uma das 10 linguagens, os dois termos mais utilizados também foram *help-wanted* e *good first issue*, indicando que a linguagem do projeto não parece estar relacionada a escolha dos rótulos.

Rótulo	Ocorrência
help wanted	404
good first issue	355
hacktoberfest	118
support	48
easy	34

Tabela 4.3. Classificação da utilização de rótulos por termos

Fonte: Autoria Própria.

Para possibilitar a observação da ocorrência dos demais termos também fora gerado a Figura 4.4. Nesta figura quanto maior a fonte do termo disposto, maior a ocorrência do mesmo nos projetos analisados.

Figura 4.4. Mapa de termos utilizados em rótulos para novatos

Fonte: Autoria Própria.

É interessante observar também que o termo *hacktoberfest* apresentou-se como o 3º termo mais utilizado, porém sabe-se que este termo está relacionado ao evento descrito na Seção 2.4, que é promovido anualmente e portanto sua utilização para indicar tarefa para novatos em outros períodos do ano pode não fazer tanto sentido. É possível que os projetos simplesmente estejam optando por não desativar o rótulo ao fim do evento e isso pode conflitar com a finalidade da prática abordada neste projeto.

4.3 Questões de Pesquisa

Para condução dessa pesquisa foram elaboradas duas questões de pesquisa. A seguir serão apresentadas as questões elaboradas, suas respectivas motivações e a metodologia executada para obter as respostas de cada uma delas.

> **QP1:** *Existe diferença estatística entre o fluxo de novatos ingressantes no período anterior e posterior a adesão da prática?*

Para esta questão, foram avaliados apenas repositórios onde fora detectado o uso de rótulos para novatos, para isso, utilizou-se o conjunto de variações de rótulos apresentada na Tabela 4.1. Por meio de *script* verificou-se para cada projeto, quais rótulos do mesmo encontravam-se entre essas variações.

Após a classificação dos projetos aderentes da prática, optou-se por aplicar um tratamento nas distribuições a serem analisadas, de modo que foram desconsiderados os dados referentes as contribuições realizadas nos 6 primeiros meses de existência dos repositórios. Essa decisão foi tomada para minimizar a possibilidade de ocorrência de mantenedores sendo contabilizados como novatos, visto que as primeiras contribuições podem ser dos criadores do projeto e não de novatos.

Para cada um dos 1060 repositórios analisados nesse trabalho foram coletados, ID da tarefa, data de submissão e autor de todas as contribuições efetuadas por novatos. A partir das datas de submissão, as contribuições foram agrupadas em uma distribuição semanal, sendo representada por um vetor de inteiros onde cada posição corresponde a uma semana de vida do projeto e seu valor descreve a quantidade de novatos ingressantes naquela semana. Essa distribuição foi divida em duas, o ponto de divisão se deu pela data de criação do rótulo para novato de cada projeto. Havendo mais de um rótulo para novato, para a divisão foi considero o mais antigo.

A fim de ilustrar melhor esta abordagem, na Figura 4.5 é apresentado um gráfico que representa a distribuição do ingresso de novatos no repositório do projeto Ansible[2], um dos selecionados para essa investigação. Optou-se pelo modelo de gráfico em linhas, pois este é o mais adequado para visualização de séries temporais. Neste gráfico o eixo horizontal representa cada semana de existência do repositório e o eixo vertical a quantidade de contribuidores que realizaram um *pull-request* pela primeira vez naquela semana. A faixa vermelha no início do gráfico representa o tratamento explicado anteriormente, que foi aplicado aos 6 primeiros meses de existência do projeto, onde foram desconsiderados os dados referentes a este período. Já a linha tracejada vermelha que corta o gráfico, representa a data onde foi verificado pela primeira vez a ocorrência de um rótulo para novato neste repositório. Esse foi o marco utilizado para determinar onde a distribuição é dividida.

[2] https://github.com/ansible/ansible

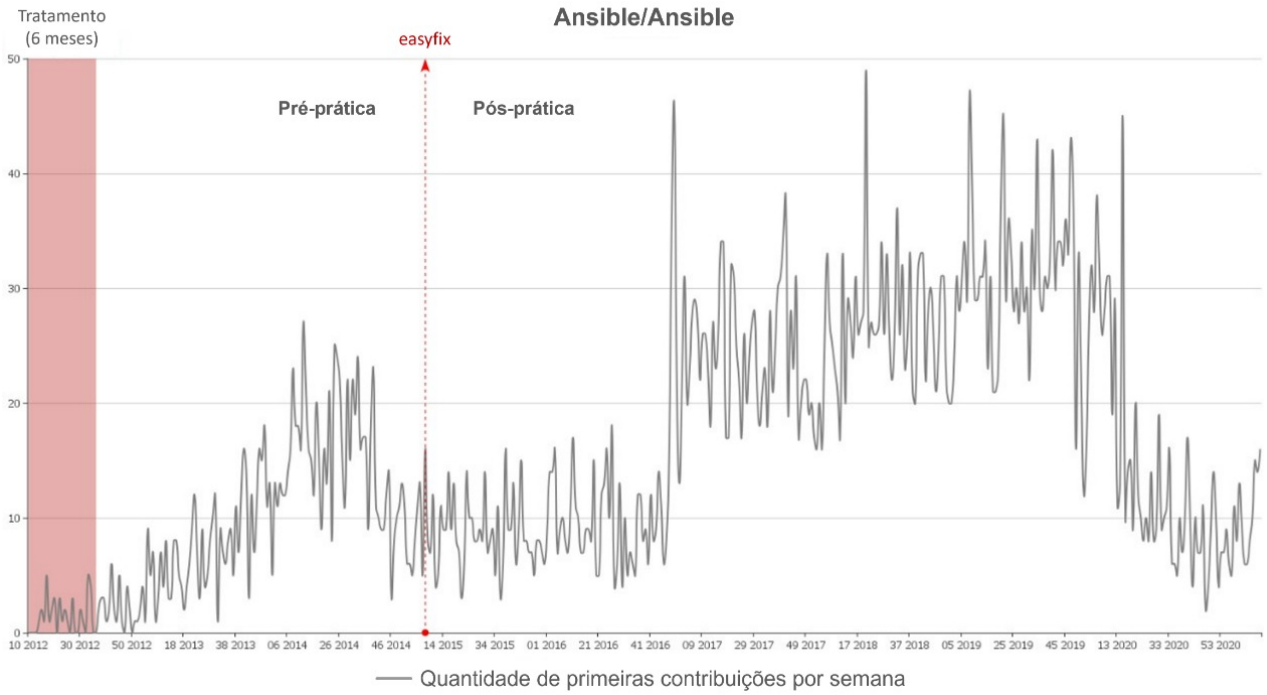

Figura 4.5. Exemplo de distribuição de ingresso de novatos - Ansible

Fonte: Autoria Própria.

Foram gerados gráficos[3] como o da Figura 4.5 para cada um dos 1060 projetos. Os gráficos resultantes também servem para verificar visualmente se os resultados obtidos pela aplicação dos testes estatísticos são coerentes e para identificar possíveis fenômenos nas amostras. Supõe-se que o período pós adoção da prática apresente maior fluxo de ingressantes novatos no projeto. A fim de validar essa hipótese primeiro verificamos se existe diferença estatística entre as distribuições representantes dos dois períodos. Quando se dispõe de uma amostra com distribuições não homogêneas, como neste caso, o teste de hipótese mais indicado é o *Mann-Whitney-Wilcoxon* (MWW) (MCELDUFF et al., 2010).

O teste MWW verifica se os valores das distribuições comparadas são diferentes com grau de confiança de 95%, para um α = 0.05, α representa o nível de significância. Se o valor-*p* (nível descritivo do teste) resultante da execução do MWW for menor do que o α, rejeita-se a *hipótese nula*, que neste contexto, pressupõe que as distribuições referentes ao ingresso de novatos antes e depois da adoção da prática são idênticas, portanto, negando-a, conclui-se que existe uma diferença estatística entre as distribuições das amostras avaliadas. Desta forma quanto menor o valor-*p*, maior a evidência de que as populações são diferentes.

O trecho de código a seguir foi desenvolvido na linguagem R e utiliza a implementação do teste de MWW provida pelo pacote *stats*[4], detalhes sobre funcionamento deste recurso estão disponíveis na documentação oficial da linguagem . Neste trecho de código a variável *treatment* representa a distribuição pós adoção da prática, visto que, a prática é o tratamento que estamos

[3] https://github.com/MateusVT/UTFPR-2021-TCC-Mateus-Torres/tree/main/Data/Results/Graphs
[4] https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/wilcox.test

investigando. Já a variável *control* representa o período pré adoção da prática, isto é, antes da aplicação do tratamento. O parâmetro *alternative* é bastante importante, pois é ele que quando atribuído com o valor "*two.sided*" define que a hipótese considera pelo teste é de que as distribuições são diferentes.

Código 1 Implementação do Teste de MWW

```
#Distribuição semanal de novatos no projeto pós adoção da prática
treatment =  project$weekly_distribuition_after
#Distribuição semanal de novatos no projeto pré adoção da prática
control =  project$weekly_distribuition_before
#Excução do teste MWW
wilcox.test(treatment, control, alternative="two.sided")
```

Complementarmente ao teste de hipótese fora aplicado o teste *Delta de Cliff* (tamanho de efeito), cuja finalidade é quantificar a magnitude da diferença entre as distribuições avaliadas, ou seja, o impacto que o tratamento teve na população (FENG; CLIFF, 2004). Este teste basicamente mede quantas vezes os valores em uma distribuição são maiores do que os valores em uma segunda distribuição e não requer nenhuma suposição sobre a extensão das duas distribuições, portanto, assim como o MWW é classificado como um teste não paramétrico (MACBETH et al., 2011).

O trecho de código a seguir também foi desenvolvido na linguagem R porém a implementação do teste é proveniente do pacote *effsize*[5]. Para a execução deste teste é importante que a ordem dos parâmetros seja respeitada. O primeiro parâmetro refere-se a distribuição com tratamento, neste caso, pós adoção da prática, já o segundo parâmetro, refere-se a distribuição antes do tratamento, ou seja, pré adoção da prática. A atribuição dos parâmetros na ordem correta é imprescindível, pois o valor do Delta pode variar dentro do intervalo de -1 á 1, e o sinal resultante indica qual das distribuições avaliadas possui os maiores valores.

Código 2 Implementação do teste *Delta de Cliff*

```
#Distribuição semanal de novatos no projeto pós adoção da prática
treatment =  project$weekly_distribuition_after
#Distribuição semanal de novatos no projeto pré adoção da prática
control =  project$weekly_distribuition_before
#Excução do teste Delta de Cliff
cliff.delta(treatment, control)
```

Considerando o Trecho de Código 2, um valor Delta positivo indica que a distribuição referente ao período posterior a adoção da prática (tratamento) apresentam maior quantidade de novos contribuidores e valores negativos o contrário disso. Para interpretar o tamanho do efeito, usaremos a escala proposta por Romano et al. (2006) aplicada ao valor do Delta (δ) obtido.

[5] https://www.rdocumentation.org/packages/effsize/versions/0.8.1/topics/cliff.delta

Relevância	Critério
insignificante	$\delta < 0.147$
baixa	$\delta < 0.33$
média	$\delta < 0.474$
alta	$\delta \geq 0.474$

Tabela 4.4. Escala para interpretação do tamanho do efeito

Fonte: (ROMANO et al., 2006).

A coluna da esquerda da Tabela 4.4 refere-se qualitativamente a relevância do efeito do delta a ser interpretado. Já a coluna adjacente apresenta as condições de classificação do delta. Analisando o resultado de ambos os testes para cada um dos projetos podemos responder esta questão de pesquisa e ter uma percepção mais refinada do impacto desta prática na atratividade dos projetos de Software Livre (SL).

QP2: *Há diferença estatística entre a quantidade de novatos dos projetos que praticam a rotulação e dos que não o fazem?*

Nesta questão de pesquisa investigamos a atratividade de dois grupos distintos de projetos, sendo eles divididos pelo critério de adoção ou não da prática de rotulação de tarefas para novatos. Diferente da **QP2** onde buscou-se compreender o impacto da prática nos projetos em si, agora, busca-se entender se os projetos que utilizam a prática podem apresentar maior atratividade de novatos por adotá-la.

A classificação dos projetos entre os dois grupos foi realizada pelo critério de presença de rótulos para novatos nos projetos, essa verificação também foi realizada por meio de *script* comparando os rótulos de cada projeto ao conjunto de rótulos mapeados, deste modo, se constatada a presença de ao menos um rótulo para novato, este era classificado como adotante da prática.

Considerando que nessa questão também foram comparados distribuições de tamanho não homogêneo, também optou-se pela utilização dos testes de hipótese não paramétricos MWW e *Delta de Cliff* para a comparação.

Nessa comparação, os dados que compuseram as distribuições que representam os dois grupos referiam-se a quantidade total de novatos que contribuíram para cada um dos projetos. Para melhor compreensão das características dessas distribuições foi gerado um gráfico *Boxplot* de ambas, este gráfico é apresentado e discutido na Seção 5.2. Na estatística, o *Boxplot*, ou diagrama de caixa, é uma maneira gráfica de representar distribuições de valores e indicada para análises comparativas de distribuições (KAMPSTRA, 2008), como é o caso abordado nesta questão de pesquisa.

A aplicação dos testes MWW e *Delta de Cliff* foram executadas da mesma maneira descrita na **QP2**, onde primeiro verificou-se com o teste MWW se existia diferença entre as duas distribuições

e depois com o *Delta de Cliff* o tamanho desta diferença. Também foram utilizadas as implementações apresentadas nos Trechos de Código 1 e 2 respectivamente. Verificando que o teste MWW resultou em valor-p < 0.05 e o *Delta de Cliff* um Delta > 0.33, leia-se: as distribuições são diferentes e a distribuição constituída pelos projetos que utilizam rótulos para novatos apresenta valores maiores, poderemos inferir que os resultados evidenciam uma maior atratividade de novatos por parte dos projetos que adotam a prática.

4.4 Seleção dos Projetos e Linguagens

Para a execução dos métodos propostos em cada uma das questões de pesquisa, realizou-se uma seleção dos repositórios mais populares disponíveis no GitHub. No contexto do GitHub a popularidade é representada pela quantidade de estrelas[6] atribuídas a um repositório (BORGES et al., 2016).

A popularidade dos repositórios foi o critério de seleção escolhido, pois estudos apontam que este fator pode estar relacionado a atratividade dos projetos (LERNER; TIROLE, 2002; ROBERTS et al., 2006; SANTOS et al., 2013). Desta forma os projetos mais populares tendem a apresentar dados mais significativos para serem analisados. Utilizando a API do GitHub para coleta dos dados conforme descrito na Seção 4.5, foram listados todos os repositórios com mais de 2500 estrelas no GitHub, totalizando aproximadamente 4000 elementos que posteriormente foram agrupados pela sua linguagem de implementação.

Após o agrupamento por linguagem, subtraiu-se repositórios de documentação e os demais foram ordenados de forma decrescente pelo critério de quantidade de estrelas. Dos grupos de repositórios por linguagem resultantes, foram selecionados apenas os referente as 10 linguagens mais populares classificadas pela aplicação *"The Top Programming Languages 2021"* publicada no *website* da IEEE.

A Figura 4.6 apresenta as linguagens que foram escolhidas, com exceção da linguagem R, que não possuía repositórios o suficiente com mais de 2500 estrelas para representá-la e, portanto, foi substituída pela próxima linguagem da ordem, o *Typescript*.

[6] Representação do interesse ou apreciação de usuários do GitHub por um projeto.

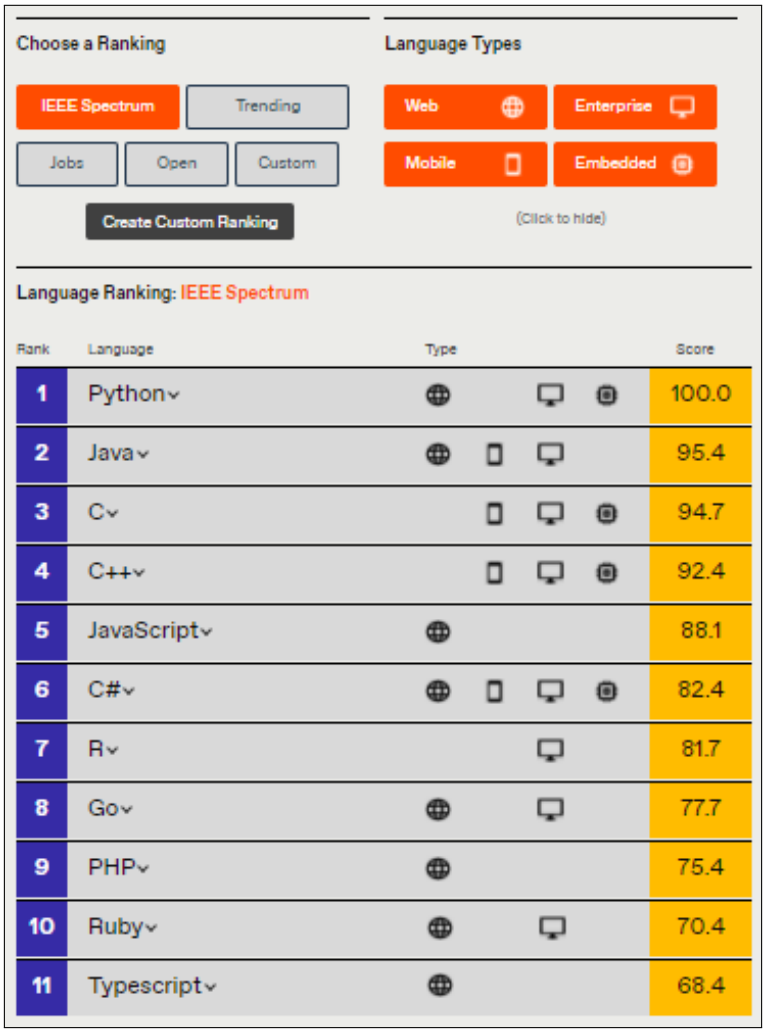

Figura 4.6. *Ranking* das linguagens mais populares 2021

Fonte: https://spectrum.ieee.org/top-programming-languages/.

Apenas os grupos de repositórios representantes das 10 linguagens mais populares foram selecionados e totalizaram 1060 repositórios de código aberto cujos dados foram coletados para os procedimentos de pesquisa descritos na Seção 4.3. A tabela a seguir apresenta a quantidade de projetos representantes para cada uma das linguagens e dados estatísticos relativos a quantidade de novatos nos projetos de cada uma delas. A partir da terceira coluna, e no contexto dos projetos selecionados, os dados referem-se à: quantidade mínima, quantidade máxima e média de novatos verificadas no conjunto de projetos de cada linguagem assim como a mediana e desvio padrão desses valores.

As informações apresentadas pela Tabela 4.5 sugerem que os projetos selecionados para esse trabalho apresentam vasta diversidade de repositórios de diferentes tamanhos, domínios e características. Uma lista com todos os repositórios coletados para esta pesquisa está disponível no GitHub[7].

[7] <https://github.com/MateusVT/UTFPR-2021-TCC-Mateus-Torres/blob/main/Data/Samples/all-repositories.json>

		Dados estatísticos relativos a quantidade de novatos				
Linguagem	**Qtd Projetos**	**Mín**	**Máx**	**Média**	**Mediana**	**Desvio Padrão**
Python	173	0	7596	570	317	893
Java	142	10	2779	388	242	395
C	118	5	2849	344	184	424
C++	179	3	4209	361	211	506
JavaScript	152	0	6828	643	295	986
C#	61	25	1895	346	249	358
Go	68	5	5000	650	407	787
PHP	84	20	5476	622	327	838
Ruby	52	75	6041	705	476	876
TypeScript	31	1	17140	1280	534	3040

Tabela 4.5. Projetos selecionados por linguagem e estatísticas relativas a quantidade de novatos - 2021

Fonte: Autoria Própria.

4.5 Coleta de Dados

Esta seção descreve como e quais foram os dados coletados para serem utilizados na metodologia proposta. Os primeiros dados a serem coletados referem-se a seleção dos repositórios que serão utilizados para a aplicação da metodologia.

Os dados referentes aos projetos mais estrelados do GitHub foram coletados por meio da execução de um script de mineração de dados implementado na linguagem **Typescript**[8], que utilizou-se da *Application Programming Interface* (API) do GitHub v3 e, por meio de requisições *Hypertext Transfer Protocol* (HTTP), que retornam um *JavaScript Object Notation* (JSON) contendo informações sobre os projetos, obteve-se uma lista dos repositórios hospedados no GitHub que possuem mais de 2500 estrelas. A requisição a seguir mostra um exemplo de consulta, por meio da API do GitHub, para obter os repositórios com quantidade de estrelas superior a 2500: <https://api.github.com/search/repositories?q=stars:>2500&s=stars&type=repositories>.

Para cada um dos repositórios selecionados, foram coletados, também por meio de script, o conjunto de todos os rótulos registrados destes projetos. Além disso, foram coletados, manualmente, todas as variações de rótulos para novatos registradas no Up-For-Grabs.

Também foram coletados de todos os repositórios, dados referentes quantidade de usuários que submeteram um *pull request* pela primeira vez (novatos), sendo este aceito ou não. Para todos os repositórios identificados como adotantes da prática, foram coletadas as datas em que foi registrado pela primeira vez a atribuição de um rótulo para novatos em suas tarefas. Para coletar esta data foi

[8] TypeScript é uma linguagem de programação de código aberto desenvolvida pela Microsoft.

necessário passar como parâmetro de busca o rótulo para novato identificado em cada um destes repositórios.

O *script* utilizado para coletar os dados descritos nessa seção foi executado na data 24/10/2021 e levou aproximadamente 72 horas para concluir o processo, o longo tempo de execução se deve a limitações de requisições impostas pela API do GitHub e pela grande quantidade de dados principalmente referentes as primeiras contribuições. Este *script* está disponível publicamente no GitHub[9]. A seguir é apresentado a estrutura e os dados que foram coletados para cada um dos projetos.

Código 3 Estrutura de dados coletadas dos projetos em formato de JSON

```
{
    id: number,
    owner: string,
    name: string,
    full_name: string
    url: string,
    language: string,
    stars: number,
    forks: number,
    subscribers: number,
    watchers: number,
    created_at: string,
    labels: string[],
    newcomer_labels: {
                        name: string,
                        created_at: string
                    }[],
    has_newcomer_labels: Boolean,
    first_contributions: {
                        login: string,
                        created_at: string,
                        issue_number?: number
                        },
    weekly_distribution: {
                        week: string,
                        dates: string[],
                        total: number
                        }[],
    weekly_distribution_before: number[],
    weekly_distribution_after: number[]
}
```

[9] https://github.com/MateusVT/UTFPR-2021-TCC-Mateus-Torres/tree/main/Scripts/git-crawler

5 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados e discutidas as análises referentes aos resultados obtidos após a execução dos procedimentos descritos na metodologia deste trabalho.

5.1 Existe diferença estatística entre o fluxo de novatos no período anterior e posterior a adesão da prática?

Nesta questão de pesquisa buscou-se verificar se a prática da utilização de rótulos para indicar tarefas recomendadas para novatos de fato tem impacto positivo na atratividade dos projetos que o fazem. Para isso, foram analisados dados relativos a atratividade do projeto antes e depois da adoção da prática.

Primeiramente foi necessário distinguir os projetos que possuíam rótulos para novatos dos que não possuíam, visto que a proposta desta questão de pesquisa envolve apenas a amostragem aderente a prática. Deste modo, foram coletadas todos os rótulos de cada um dos 1060 projetos e esses conjuntos de rótulos foram comparados um a um com a amostra apresentada na Tabela 4.1. Quando constatado a presença de um ou mais rótulos do projeto na amostra de rótulos para novatos o projeto era classificado como aderente da prática, vale salientar que a comparação desconsiderou *case sensitive*[1], ou seja, uma comparação entre o termo *"good first issue"* e *"Good First Issue"* seria positiva por exemplo. Nessa verificação alguns projetos apresentaram presença de mais de um rótulo para novato em seu conjunto de rótulos.

Conforme mencionado na metodologia, os procedimentos foram desempenhados para o **Cenário i** e **Cenário ii**, portanto, essa verificação também foi executada novamente nas amostras após a remoção dos termos descritos na Tabela 4.2, de modo a representar o **Cenário ii**.

No **Cenário ii** projetos onde anteriormente haviam sido identificados apenas um rótulo de novato, e este era um termo similar a *help-wanted* passaram a ser classificados como não aderentes da prática. Um exemplo onde isso ocorreu é o projeto *Cinder*[2], que passou ser considerado um projeto não praticante para o **Cenário ii**. Já projetos onde haviam sido detectados outros rótulos para novatos passou-se a considerar o segundo rótulo mais antigo como representante do mesmo, como é o caso dos projetos: *Openssl*[3], *Babylon.js*[4] e *Rubygems*[5].

Deste modo, dos 1060 projetos para o **Cenário i** tivemos um total de 661 (62.4%) projetos classificados como aderentes a prática e para o **Cenário ii**, 544 (51.4%) projetos.

[1] Sensível a letras maiúsculas e minúsculas
[2] https://github.com/cinder/cinder
[3] https://github.com/openssl/openssl
[4] https://github.com/babylonJS/babylon.js
[5] https://github.com/rubygems/rubygems

Após a definição da amostragem para execução dos testes, se fez necessário identificar o momento em que o rótulo foi utilizado pela primeira vez no dado repositório de forma que fosse possível separar a distribuição de ingresso de novato em duas distribuições representando os dois períodos do projeto, o pré e o pós adesão da prática. Para os projetos onde fora identificado mais de um rótulo para novato, definiu-se o mais antigo como represente do projeto. Posteriormente coletamos a data de primeira incidência deste rótulo no projeto. A partir desta data dividiu-se em duas a distribuição de ingresso semanal de novatos do projeto.

Neste momento percebeu-se que alguns projetos começaram a utilizar os rótulos para novatos nos 6 primeiros meses de sua existência no GitHub, o que impossibilitou a divisão dos períodos visto que os dados referente a este período foram desconsiderados. No **Cenário i**, 210 projetos apresentaram essa característica e no **Cenário ii**, 125, assim sendo, estes projetos foram desconsiderados na execução dos testes estatísticos, restando 451 e 419 projetos a serem analisados para cada cenário respectivamente.

Em seguida os testes estatísticos não-paramétricos MWW e o *Delta de Cliff* (tamanho do efeito) foram executados, por meio de *script*[6], para cada um dos projetos restantes e para ambos os cenários, de modo que a distribuição pré-rótulo representa o grupo de controle e a pós-rótulo de tratamento.

Do total de projetos testados para o **Cenário i**, 412 apresentaram diferença nas distribuições e no **Cenário ii**, 419, isto é, o valor resultante do teste MWW, foi $p < 0{,}05$. Em relação ao teste *Delta de Cliff* executado posteriormente para análise de tamanho de efeito é necessário considerar o sinal aritmético que acompanha o *Delta* resultante, este sinal indica qual das distribuições apresenta valores maiores. Neste contexto um *Delta* negativo indica que o período anterior a adoção da prática (controle) apresentou maior atratividade de novatos. 62 projetos no **Cenário i** e 54 no **Cenário ii** apresentaram *Delta* negativo. Destes grupos, aproximadamente metade dos projetos apresentaram diferença significativa indicando menor atratividade após a adesão a prática. Um dos projetos que apresentou esse resultado foi o *firefox-devtools/debugger*[7], na Figura 5.1 podemos observar uma decrescente na contagem de novatos ingressantes após a utilização de rótulo para novatos.

[6] https://github.com/MateusVT/UTFPR-2021-TCC-Mateus-Torres/blob/main/Scripts/executeTests.py
[7] https://github.com/firefox-devtools/debugger

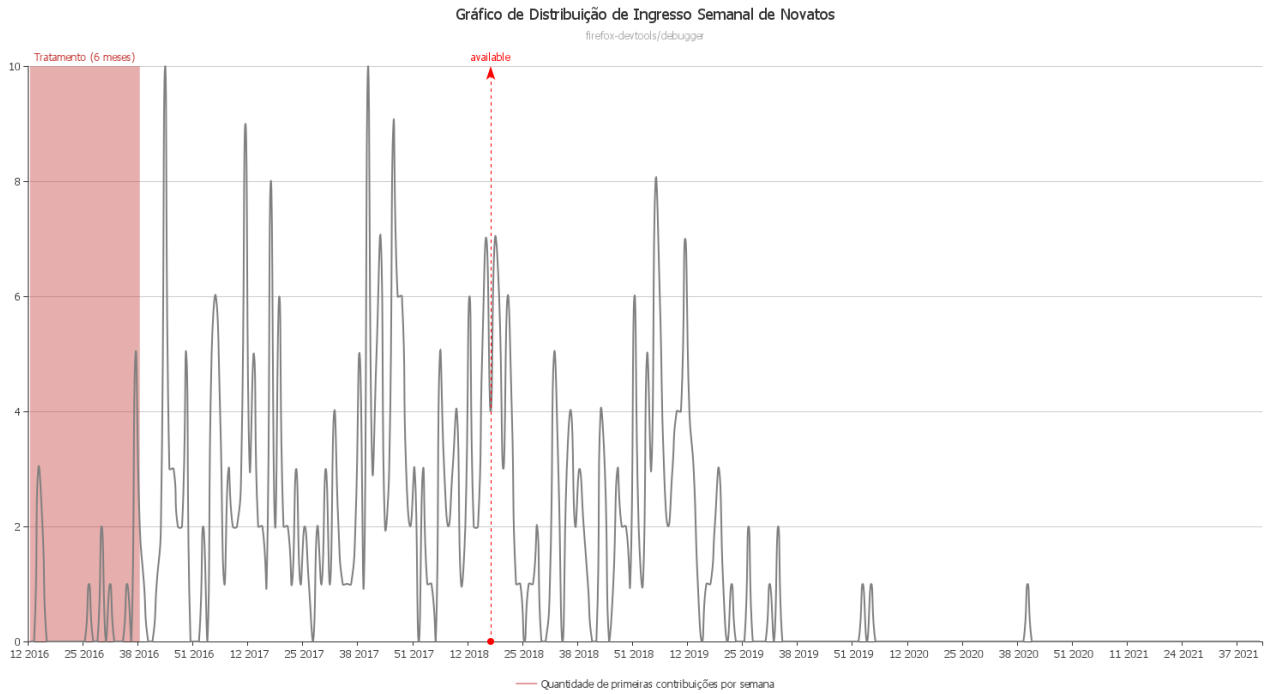

Figura 5.1. Exemplo de distribuição de ingresso de novatos - Firefox Debugger

Fonte: Autoria Própria.

Visando compreender o motivo pelo qual esses projetos passaram a apresentar menor atratividade após a utilização da prática, buscou-se mais informações sobre esses casos em específico em seus repositórios no GitHub. Constatou-se que em boa parte dos casos onde houve queda na atratividade, os projetos tiveram seus repositórios migrados para outro serviço de hospedagem ou foram desativados pelo seu proprietário, como é o caso do projeto *firefox-devtools/debugger*. Sendo assim, essa diminuição da atratividade, não parece estar relacionada a adoção da prática.

Subtraindo-se os casos com *Delta* negativo e agrupando estes projetos por cenário e pelo tamanho do efeito, obtemos a seguinte composição:

Relevância	Repositórios (Cenário i)	Repositórios (Cenário ii)
alta	110	106
média	66	60
baixa	112	97
insignificante	47	43

Tabela 5.1. Resultado dos testes estatísticos por cenário

Fonte: Autoria Própria.

Com base nos resultados apresentados na Tabela 5.1 constatou-se que em ambos os cenários, aproximadamente 63% dos projetos apresentaram uma diferença significativa (baixa, média ou alta) entre as distribuições. Deste modo, podemos inferir que a utilização ou não de termos semelhantes a

help-wanted não resultou em diferença expressiva na aplicação dos testes, mesmo considerando as classificações por relevância.

Para entender se a linguagem de implementação do projeto pode ter alguma influência nos resultados obtidos, também foram geradas tabelas referentes aos resultados dos testes estatísticos aplicando agrupamento por linguagem e para cada um dos cenários. Na Tabela 5.2, são exibidos os valores para cada uma das linguagens, ordenadas de forma decrescente por sua popularidade, e referentes a: quantidade de projetos praticantes, quantidade de projetos que foram descartados na avaliação devido impossibilidade de separação das distribuições e quantidade de projetos classificados para cada um dos tamanhos de efeitos previstos pela escala de Romano et al. (2006).

Considerando que as linguagens dispõem de quantidade de amostras representantes de tamanho diferente, optou-se por ao lado de cada valor unitário exibir o valor em porcentagem que representa a razão deste valor para o total de projetos praticantes, deste modo pode-se avaliar com mais facilidade os resultados apresentados.

	projetos praticantes	descartados	insignificante	baixa	média	alta
Python	115 de 173 (66%)	29 (25%)	18 (16%)	27 (23%)	11 (10%)	30 (26%)
Java	72 de 142 (51%)	22 (31%)	20 (28%)	12 (18%)	08 (11%)	09 (13%)
C	57 de 118 (48%)	15 (26%)	14 (25%)	07 (12%)	07 (12%)	14 (25%)
C++	103 de 179 (58%)	35 (34%)	23 (22%)	25 (24%)	08 (8%)	12 (12%)
JavaScript	96 de 152 (63%)	33 (34%)	22 (23%)	18 (19%)	10 (10%)	13 (14%)
C#	49 de 61 (80%)	21 (43%)	10 (21%)	07 (14%)	08 (16%)	03 (06%)
Go	53 de 68 (78%)	23 (43%)	06 (11%)	06 (11%)	08 (15%)	10 (19%)
PHP	58 de 84 (69%)	14 (24%)	14 (24%)	12 (21%)	03 (5%)	15 (26%)
Ruby	35 de 52 (67%)	06 (17%)	08 (23%)	10 (29%)	04 (11%)	07 (20%)
Typescript	23 de 31 (74%)	12 (52%)	03 (13%)	0 (0%)	05 (22%)	03 (13%)

Tabela 5.2. Resultados dos testes estatísticos por linguagem - **Cenário i**

Fonte: Autoria Própria.

Os dados da coluna "projetos praticantes" na Tabela 5.2 podem ser utilizados para prover uma percepção da adoção da prática por projetos de cada linguagem. Podemos verificar que, com exceção da linguagem *C*, todas as linguagens apresentam uma taxa de adoção da prática superior a 50%. Os valores apresentados nessa coluna também indicam que parece não haver uma relação entre a popularidade das linguagens e a adoção da prática em seus projetos, visto que as 5 linguagens menos populares da classificação apresentaram valor maiores do que as 5 mais populares. Entretanto, é interessante observar que, as 5 linguagens menos populares são também em média mais recentes do que as demais (LÉVÉNEZ, 2004), então é possível que exista uma relação entre tecnologias mais modernas e a adesão a prática.

Ademais, observando as colunas referentes ao tamanho de efeito resultantes dos projetos de cada linguagem, não foi constatado valores discrepantes entre os elementos, de modo que, não parece haver relação entre a linguagem e o tamanho do efeito da adoção da práticas em seus projetos. Mesmo considerando a soma dos efeitos significativos (baixo, médio e alto) de cada linguagens,

não foi observado, se comparado a média entre eles, nenhum valor de destaque, para alguma das linguagens.

Na Tabela 5.3 temos a exibição dos mesmo dados porém referentes aos valores resultantes dos testes estatísticos aplicados no **Cenário ii**. Comparando os dados apresentados na primeira coluna desta tabela com a mesma coluna da Tabela 5.2, é possível verificar uma pequena diminuição na quantidade de projetos adotantes da prática, em todas as linguagens. Podemos inferir, que desconsiderando rótulos relacionados ao termo *help-wanted* temos menos projetos com a presença de rótulos para novatos em seus repositórios.

Contudo, se compararmos a coluna referente aos "projetos descartados", também observamos uma pequena diminuição nos valores da Tabela 5.3. Sabendo que vários projetos apresentaram outros rótulos além do similares a *help-wanted* em seu conjunto de rótulos, e que foi possível efetuar a divisão dos períodos pré e pós adesão da prática em mais projetos no **Cenário ii**, isto pode indicar que alguns projetos utilizavam os rótulos *help-wanted* desde o início do projeto, não com o propósito almejado pela prática, e posteriormente adotaram outro rótulo para essa finalidade.

	projetos praticantes	**descartados**	**insignificante**	**baixa**	**média**	**alta**
Python	102 de 173 (59%)	18 (18%)	21 (21%)	26 (25%)	09 (9%)	28 (27%)
Java	55 de 142 (39%)	13 (23%)	16 (29%)	11 (20%)	07 (13%)	08 (15%)
C	40 de 118 (34%)	07 (18%)	08 (20%)	06 (15%)	08 (20%)	11 (27%)
C++	81 de 179 (45%)	22 (27%)	19 (23%)	23 (29%)	07 (09%)	10 (12%)
JavaScript	77 de 152 (51%)	19 (25%)	23 (30%)	14 (18%)	10 (13%)	11 (14%)
C#	41 de 61 (67%)	14 (34%)	11 (27%)	05 (12%)	08 (20%)	03 (07%)
Go	49 de 68 (72%)	13 (27%)	09 (18%)	06 (12%)	09 (18%)	12 (25%)
PHP	53 de 84 (63%)	10 (19%)	15 (28%)	12 (23%)	03 (06%)	13 (24%)
Ruby	29 de 52 (56%)	03 (10%)	07 (24%)	08 (28%)	04 (14%)	07 (24%)
Typescript	18 de 31 (58%)	07 (39%)	02 (11%)	0 (0%)	03 (17%)	06 (33%)

Tabela 5.3. Resultados dos testes estatísticos por linguagem - **Cenário ii**

Fonte: Autoria Própria.

Considerando as colunas referente ao efeito da prática nos projetos na Tabela 5.3, também não se destacaram nenhum dos valores seja do ponto de vista de linguagem ou entre os cenários.

De modo geral o agrupamento por linguagem não destacou grandes impactos na relação entre a adoção da prática e as linguagens, visto que a diferença entre a média dos efeitos significativos resultantes dos testes estatísticos para cada linguagem não é expressiva. Todavia, os resultados apresentados na Tabela 5.1 evidenciaram que uma parcela considerável dos projetos apresentaram aumento significativo em relação a sua atratividade de novatos após a adoção da prática, independente do cenário considerado.

5.2 Há diferença estatística entre o fluxo de entrada de novatos dos projetos que praticam a rotulação e dos que não o fazem?

Para entender se há diferença na atratividade de novatos entre projetos que aderem a prática de rotulação de tarefas para novatos e projetos que não o fazem, foram executadas comparações também aplicando os testes estatísticos não-paramétricos MWW e *Delta de Cliff*. Entretanto, diferente da **QP1** as duas distribuições comparadas nesta questão de pesquisa são constituídas pelo total de primeiras contribuições de cada projeto desses dois grupos.

A tabela a seguir apresenta métricas estatísticas das duas distribuições comparadas para o **Cenário i**. Podemos observar que mediana de novatos em projetos aderentes a prática é consideravelmente maior do que a mediana dos que não o fazem. Sabendo que o teste *Delta de Cliff* estima a probabilidade de uma observação aleatoriamente selecionada de um grupo ser maior do que uma observação aleatoriamente selecionada de outro grupo, menos a probabilidade reversa (CLIFF, 2014), espera-se um *Delta* tendendo a distribuição possuinte do maior valor de mediana, neste caso, a distribuição dos projetos com rótulos para novatos. Portanto com base na Tabela 5.4 espera-se um *Delta* positivo.

		Dados estatísticos relativos a quantidade de novatos				
Critério	**Projetos**	**Mín**	**Máx**	**Média**	**Mediana**	**Desvio Padrão**
Com rótulo para novatos	661	5	17140	624	337	1042
Sem rótulo para novatos	399	0	4385	346	197	492

Tabela 5.4. Estatísticas das distribuições - **Cenário i**

Fonte: Autoria Própria.

Para uma melhor visualização dos dados referentes as estatísticas das distribuições também foram gerados gráficos *boxplot* das mesmas.

No *boxplot* se percebe as medidas de quartis, mediana, amplitude, além de nos permitir identificar muito bem os elementos *outliers*[8], entretanto neste caso as distribuições apresentaram diversas ocorrências de *outliers* o que tornou inviável a visualização das demais informações no gráfico *boxplot*, deste modo optou-se por desabilitar a exibição de *outliers* na plotagem do gráfico apresentado na Figura 5.2.

[8] *outliers* são dados que se diferenciam drasticamente de todos os outros, são pontos fora da curva normal

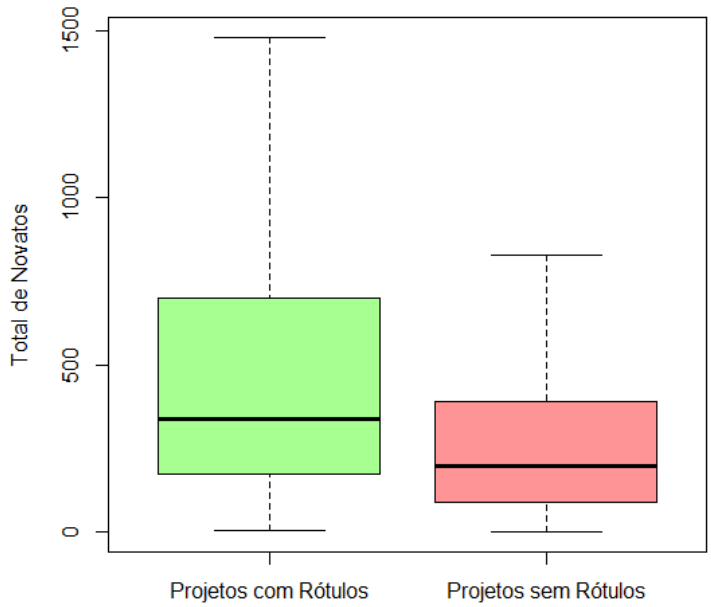

Figura 5.2. *Boxplot* da distribuição de ingresso de novatos - **Cenário i**
Fonte: Autoria Própria.

O Figura 5.2 reforça visualmente a diferença entre as distribuições referentes aos dois grupos. Percebe-se também que mesmo após a remoção dos *outliers* os valores máximos dos projetos praticantes são consideravelmente superiores ao do outro grupo. Assim como temos uma relação entre mediana e o resultado do teste *Delta de Cliff*, que foi mencionada anteriormente, por meio da Figura 5.2 podemos validar o resultado esperado no teste MWW, visto que este gráfico indica que existe diferença entre as distribuições. Deste modo, tanto os dados da Tabela 5.4 quanto a Figura 5.2 são auxiliares para validarmos os resultados obtidos pelos testes estáticos MWW e *Delta de Cliff*.

A Tabela 5.5 por sua vez dispõe os dados estatísticos referentes as distribuições para o **Cenário ii**. Podemos observar que apesar de a quantidade de projetos classificados como praticantes ter diminuído, o que é um efeito esperado para esse cenário, não observamos grandes diferenças nos demais dados estatísticos se comparados aos do **Cenário i**. A semelhança entre os dados de ambos os cenários corrobora com a observação feita na discussão dos resultados da **QP1**, de que, desconsiderar rótulos relacionados ao termo *help-wanted* não resulta em grande impacto nas análises propostas neste trabalho.

		Dados estatísticos relativos a quantidade de novatos				
Critério	**Projetos**	**Mín**	**Máx**	**Média**	**Mediana**	**Desvio Padrão**
Com rótulo para novatos	544	5	17140	665	369	1069
Sem rótulo para novatos	516	0	6828	366	198	606

Tabela 5.5. Estatísticas das distribuições - **Cenário ii**

Fonte: Autoria Própria.

Comparando os gráficos das Figuras 5.2 e 5.3 fica ainda mais evidente a semelhança estatísticas entre as distribuições avaliadas em ambos os cenários. Deste modo, espera-se que o os valores resultantes dos testes estatísticos em ambos os cenários apresentem valores similares.

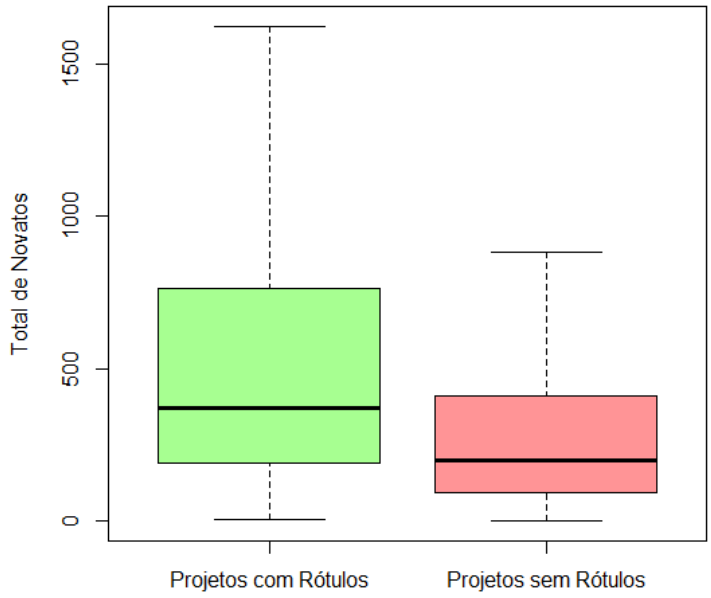

Figura 5.3. *Boxplot* de distribuição de ingresso de novatos - **Cenário ii**

Fonte: Autoria Própria.

Finalmente, executando os testes estáticos MWW e *Delta de Cliff*, obtivemos os resultados apresentados na Tabela 5.6. O p-*value* <0.05 resultante do teste MWW comprova que existe diferença estatística entre as distribuições avaliadas em ambos os cenários. Conforme discutido anteriormente, esse resultado condiz com o esperado a partir da análise das Figuras 5.2 e 5.3.

Seguindo para o teste *Delta de Cliff*, obtivemos os valores *Delta* 0.29 e 0.34 para o **Cenário i** e **Cenário ii** respectivamente. Apesar desses valores serem classificados em diferentes níveis da escala de Romano et al. (2006), são valores bastante próximos, o que por sua vez também condiz com o esperado se considerado as medianas das distribuições comparadas em ambos os cenários. Podemos observar pelas Tabelas 5.4 e 5.5 que as medianas das distribuições do **Cenário i** apresentam valores mais próximos, portanto, é coerente que o *Delta* resultante seja inferior.

	p-*value*	Delta	Relevância
Cenário i	<0.05	0.29	pequeno
Cenário ii	<0.05	0.34	médio

Tabela 5.6. Resultados dos testes estatísticos por cenário

Fonte: Autoria Própria.

Ambos os valores de *Delta* apresentados na Tabela 5.6 são positivos, o que significa que independentemente do cenário avaliado a utilização de rótulos para indicar tarefas para novatos teve efeito significativo. Isto é, projetos que adotam a prática apresentaram maior quantidade de novatos em relação aos que não o fazem.

5.3 Ameaças à Validade

A seguir são discutidas as principais ameaças à validade identificadas durante a elaboração da metodologia proposta nesse trabalho.

Mapeamento Manual de Rótulos para Novatos e Dependência do *Up-For-Grabs*

Como discutido na Seção 4.2, mesmo utilizando o conjunto de rótulos coletados do *Up-For-Grabs* como base, é possível que tenhamos elegido erroneamente algum dos rótulos, produzindo assim um viés nos resultados obtidos. O levantamento de variação de rótulos exclusivamente baseado no projeto *Up-For-Grabs* também pode ter limitado nossa análise visto que outros projetos surgiram com a mesma missão do *Up-For-Grabs* e podem conter outras variações a serem consideradas. O mapeamento manual de rótulos acrescentou outros termos a amostra apresentada na Tabela 4.1, porém, ainda sim é provável que alguns termos não tenham sido considerados. O único meio de garantir qual é o rótulo praticado por um projeto para indicar tarefas para novato, seria obtendo uma confirmação de seus mantenedores, o que seria impraticável dado o prazo deste trabalho.

Múltiplos Rótulos para Novatos em um Projeto

Como mencionado na Seção 4.2, durante a análise manual do conjunto de rótulos dos repositórios, foi identificado em alguns deles a ocorrência de múltiplos rótulos com a finalidade de indicar tarefas para novatos. Isso pode comprometer os testes estatísticos propostos na **QP1**, visto que, para dividir a distribuição entre os períodos anterior e posterior ao uso da prática é necessário saber a data em que verificou-se pela primeira vez o uso de um rótulo para novatos. Deste modo, havendo mais de um rótulo em um dado projeto, foi verificado qual dos rótulos é o mais antigo e este foi utilizado como oficial.

6 CONCLUSÕES

Nesta seção será apresentado um resumo deste estudo, focando nos resultados obtidos e os relacionando as questões de pesquisa propostas.

As investigações conduzidas neste trabalho possibilitaram uma melhor compreensão de alguns dos elementos associados a prática de utilização de rótulos para indicar tarefas a novatos. Iniciando pela identificação dos rótulos utilizados para essa prática e da prevalência da adoção da mesma em projetos de Software Livre (SL), percebemos que a variação de termos utilizados para a finalidade desta prática é bastante abundante, conforme pode ser verificado na Tabela 4.1. Alguns desses termos também não parecem ser intuitivamente relacionados a termos para novatos, deste modo campanhas como a do GitHub que sugerem termos padrões para essa finalidade podem trazer benefícios para a eficiência da prática visto que facilitam a identificação do rótulo por parte do novato.

Nesta contexto também percebeu-se que os dois termos recomendados pelo GitHub para essa finalidade, *"good first issue"* e *"help wanted"*, vem sendo majoritariamente utilizados pelos projetos, entretanto em 71% projetos que possuíam um rótulo com termo *help wanted* também foi identificado a presença de outros rótulos para novatos, o que pode indicar que os projetos não estão utilizando esse rótulo exclusivamente para este fim, de modo que cabem maiores investigações sobre este assunto.

Tratando-se da **QP1**, descobrimos que uma parcela considerável dos projetos utilizam rótulos para novatos, alguns desde sua concepção. Do total de projetos analisados neste trabalho, mais de 50% apresentaram rótulos para novatos, e se considerado o termo *help-wanted* mais de 60%. Para os projetos adotantes da prática os testes estatísticos evidenciaram que, após adoção da prática, 63% deles tiveram um aumento significativo na quantidade semanal de novatos que submeteram uma contribuição. Esse resultado destaca que essa prática realmente tem um grande potencial para melhorar a atratividade dos projetos de SL.

Quanto a última questão de pesquisa, fora comparado a quantidade total de novatos de projetos praticantes e não praticantes, a fim de verificar se a prática pode ter influenciado positivamente na atratividade desses projetos. Os resultados apontaram que projetos praticantes apresentam maior quantidade de novatos em relação aos não praticantes.

Deste modo, unindo os resultados obtidos pela **QP1** e **QP2**, obtivemos evidências de que a prática pode influenciar positivamente na atratividade de projetos de SL, sendo assim, deveria ser incentivada e divulgada pela comunidade.

Para trabalhos futuros, seria interessante entender se existe algum motivo que possa estar limitando a eficiência da prática em alguns projetos, por exemplo, a falta de divulgação ou orientação em relação ao seu uso e principalmente para identificação dos rótulos para novatos. Também pode-se investigar se outras características de um projeto podem potencializar o efeito da prática.

REFERÊNCIAS

ALDERLIESTEN, Jan Willem David; ZAIDMAN, Andy. An initial exploration of the "good first issue" label for newcomer developers. In: IEEE. **2021 IEEE/ACM 13th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)**. [S.l.], 2021. p. 117–118.

BITZER, Jürgen; SCHRETTL, Wolfram; SCHRÖDER, Philipp JH. Intrinsic motivation in open source software development. **Journal of Comparative Economics**, Elsevier, v. 35, n. 1, p. 160–169, 2007.

BORGES, Hudson; HORA, Andre; VALENTE, Marco Tulio. Understanding the factors that impact the popularity of github repositories. In: IEEE. **2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)**. [S.l.], 2016. p. 334–344.

CABOT, Jordi; IZQUIERDO, Javier Luis Cánovas; COSENTINO, Valerio; ROLANDI, Belén. Exploring the use of labels to categorize issues in open-source software projects. In: 22nd International Conference on 2015 IEEE. **Software Analysis, Evolution and Reengineering (SANER)**. [S.l.], 2015. p. 550–554.

CLIFF, Norman. **Ordinal methods for behavioral data analysis**. [S.l.]: Psychology Press, 2014.

CUBRANIC, Davor; MURPHY, Gail C; SINGER, Janice; BOOTH, Kellogg S. Hipikat: A project memory for software development. **IEEE Transactions on Software Engineering**, IEEE, v. 31, n. 6, p. 446–465, 2005.

DIAS, Luiz Felipe; STEINMACHER, Igor; WIESE, Igor; PINTO, Gustavo; COSTA, Daniel Alencar da; GEROSA, Marco. Uma análise preliminar de projetos de software livre que migraram para o GitHub. **Sociedade Brasileira de Computação–SBC**, p. 65, 2016.

DIAS, Luiz Felipe Fronchetti. **Um estudo exploratório sobre indicadores de receptividade em projetos de software livre**. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2017.

FENG, Du; CLIFF, Norman. Monte carlo evaluation of ordinal d with improved confidence interval. **Journal of Modern Applied Statistical Methods**, v. 3, n. 2, p. 6, 2004.

FERRAZ, Nelson Corrêa de Toledo. Vantagens estratégicas do software livre para o ambiente corporativo. **Monografia em Master Business Information Systems, São Paulo, Centro de Ciências Exatas e Econômicas, Pontifícia Universidade Católica, Brasil**, 2002.

FSF. **What is free software?** 2004. Acesso em: 07 nov. 2017. Disponível em: http://www.fsf.org/about/what-is-free-software.

GHOSH, Rishab Aiyer. Interviews with linus torvalds: What motivates software developers. **First Monday**, v. 3, n. 2, 1998.

GHOSH, Rishab Aiyer. Understanding free software developers: Findings from the floss study. **Perspectives on free and open source software**, Cambridge MA: MIT Press, p. 23–46, 2005.

GOUSIOS, Georgios; PINZGER, Martin; DEURSEN, Arie van. An exploratory study of the pull-based software development model. In: ACM. **Proceedings of the 36th International Conference on Software Engineering**. [S.l.], 2014. p. 345–355.

HANN, Il-Horn; ROBERTS, Jeff; SLAUGHTER, Sandra; FIELDING, Roy. Why do developers contribute to open source projects? first evidence of economic incentives. In: **2nd Workshop on Open Source Software Engineering, Orlando, FL**. [S.l.: s.n.], 2002.

HARS, Shaosong Ou Alexander. Working for free? motivations for participating in open-source projects. **International journal of electronic commerce**, Taylor & Francis, v. 6, n. 3, p. 25–39, 2002.

HERTEL, Guido; NIEDNER, Sven; HERRMANN, Stefanie. Motivation of software developers in open source projects: an internet-based survey of contributors to the linux kernel. **Research policy**, Elsevier, v. 32, n. 7, p. 1159–1177, 2003.

HUANG, Yuekai; WANG, Junjie; WANG, Song; LIU, Zhe; WANG, Dandan; WANG, Qing. Characterizing and predicting good first issues. In: **Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)**. [S.l.: s.n.], 2021. p. 1–12.

KAMPSTRA, Peter. Beanplot: A boxplot alternative for visual comparison of distributions. **Journal of statistical software**, v. 28, n. 1, p. 1–9, 2008.

KROGH, Georg Von; SPAETH, Sebastian; LAKHANI, Karim R. Community, joining, and specialization in open source software innovation: a case study. **Research Policy**, Elsevier, v. 32, n. 7, p. 1217–1241, 2003.

LERNER, Josh; TIROLE, Jean. Some simple economics of open source. **The journal of industrial economics**, Wiley Online Library, v. 50, n. 2, p. 197–234, 2002.

LÉVÉNEZ, Éric. History of programming languages. **oreilly. com**, 2004.

MACBETH, Guillermo; RAZUMIEJCZYK, Eugenia; LEDESMA, Rubén Daniel. Cliff's delta calculator: A non-parametric effect size program for two groups of observations. **Universitas Psychologica**, Pontificia Universidad Javeriana, v. 10, n. 2, p. 545–555, 2011.

MCELDUFF, Fiona; CORTINA-BORJA, Mario; CHAN, Shun-Kai; WADE, Angie. When t-tests or wilcoxon-mann-whitney tests won't do. **Advances in physiology education**, American Physiological Society Bethesda, MD, v. 34, n. 3, p. 128–133, 2010.

MEIRELLES, Paulo; SANTOS, Carlos; MIRANDA, Joao; KON, Fabio; TERCEIRO, Antonio; CHAVEZ, Christina. A study of the relationships between source code metrics and attractiveness in free

software projects. In: IEEE. **2010 Brazilian Symposium on Software Engineering (SBES)**. [S.l.], 2010. p. 11–20.

NETMARKETSHARE. **Desktop Operating System Market Share**. 2021. Acesso em: 18 nov. 2021. Disponível em: https://www.netmarketshare.com/operating-system-market-share.aspx.

OSI, The Open Source Initiative. **History of the OSI**. 1999. Acesso em: 03 nov. 2017. Disponível em: https://opensource.org/history.

PINTO, Gustavo; STEINMACHER, Igor; GEROSA, Marco Aurélio. More common than you think: An in-depth study of casual contributors. In: 23rd International Conference on 2016 IEEE. **Software Analysis, Evolution, and Reengineering (SANER)**. [S.l.], 2016. v. 1, p. 112–123.

ROBERTS, Jeffrey A; HANN, Il-Horn; SLAUGHTER, Sandra A. Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache projects. **Management science**, INFORMS, v. 52, n. 7, p. 984–999, 2006.

ROMANO, J.; KROMREY, J.D.; CORAGGIO, J.; SKOWRONEK, J. Appropriate statistics for ordinal level data: Should we really be using t-test and Cohen'sd for evaluating group differences on the NSSE and other surveys? In: **Annual meeting of the Florida Association of Institutional Research**. [S.l.: s.n.], 2006. p. 1–3.

SANTOS, Carlos; KUK, George; KON, Fabio; PEARSON, John. The attraction of contributors in free and open source software projects. **The Journal of Strategic Information Systems**, Elsevier, v. 22, n. 1, p. 26–45, 2013.

SILVA, Danilo Augusto; BONIFÁCIO, Dhyego Palácios; MAGRINI, Márcio Luiz; BATISTA, Murillo Rehder. Software livre: direitos, deveres e efeitos para a sociedade. **USP, Maio**, 2010.

SILVA, Jefferson O; WIESE, Igor S; STEINMACHER, Igor; GEROSA, Marco A. Students' engagement in open source projects: An analysis of google summer of code. In: ACM. **Proceedings of the 31st Brazilian Symposium on Software Engineering**. [S.l.], 2017. p. 224–233.

STALLMAN, Richard M. What is free software. **Free Society: Selected Essays of**, p. 23, 2002.

STEINMACHER, Igor; CONTE, Tayana Uchôa; GEROSA, Marco Aurélio. Understanding and supporting the choice of an appropriate task to start with in open source software communities. In: IEEE. **System Sciences (HICSS), 2015 48th Hawaii International Conference on**. [S.l.], 2015. p. 5299–5308.

STEINMACHER, Igor; WIESE, Igor; CHAVES, Ana Paula; GEROSA, Marco Aurélio. Why do newcomers abandon open source software projects? In: IEEE. **Cooperative and Human Aspects of Software Engineering (CHASE), 2013 6th International Workshop on**. [S.l.], 2013. p. 25–32.

STEINMACHER, Igor; WIESE, Igor Scaliante; CONTE, Tayana; GEROSA, Marco Aurélio; REDMILES, David. The hard life of open source software project newcomers. In: ACM. **Proceedings of the 7th**

international workshop on cooperative and human aspects of software engineering. [S.l.], 2014. p. 72–78.

TAN, Xin; ZHOU, Minghui; SUN, Zeyu. A first look at good first issues on github. In: **Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering**. [S.l.: s.n.], 2020. p. 398–409.

WANG, Jianguo; SARMA, Anita. Which bug should i fix: helping new developers onboard a new project. In: ACM. **Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering**. [S.l.], 2011. p. 76–79.

WOLFF-MARTING, Vincent; HANNEBAUER, Christoph; GRUHN, Volker. Patterns for tearing down contribution barriers to floss projects. In: IEEE. **Intelligent Software Methodologies, Tools and Techniques (SoMeT), 2013 IEEE 12th International Conference on**. [S.l.], 2013. p. 9–14.

YU, Yue; WANG, Huaimin; YIN, Gang; WANG, Tao. Reviewer recommendation for pull-requests in github: What can we learn from code review and bug assignment? **Information and Software Technology**, Elsevier, v. 74, p. 204–218, 2016.