newcommand* as robustness is handled now at top level in `fmtcount.sty`, and we don't need long macros. Concerned macros are @numberstringMportuges, @numberstringFportuges, @NumberstringMportuges, @NumberstringFportuges, @ordinalstringMportuges, @ordinalstringFportuges, @OrdinalstringMportuges, and @OrdinalstringFportuges.

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUIZ FELIPE FRONCHETTI DIAS

UM ESTUDO EXPLORATÓRIO SOBRE INDICADORES DE RECEPTIVIDADE EM PROJETOS DE SOFTWARE LIVRE

MONOGRAFIA

CAMPO MOURÃO
2017

LUIZ FELIPE FRONCHETTI DIAS

UM ESTUDO EXPLORATÓRIO SOBRE INDICADORES DE RECEPTIVIDADE EM PROJETOS DE SOFTWARE LIVRE

Trabalho de Conclusão de Curso de graduação apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso de Bacharelado em Ciência da Computação do Departamento Acadêmico de Computação da Universidade Tecnológica Federal do Paraná, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador:
Prof. Dr. Igor Scaliante Wiese

Coorientadores:
Prof. Dr. Igor Steinmacher
Prof. Dr. Gustavo Pinto

CAMPO MOURÃO
2017

Resumo

Dias, Luiz Felipe Fronchetti. Um estudo exploratório sobre indicadores de receptividade em projetos de software livre. 2017. 41. f. Monografia (Curso de Bacharelado em Ciência da Computação), Universidade Tecnológica Federal do Paraná. Campo Mourão, 2017.

Projetos de software livre necessitam de contribuições voluntárias para se manterem ativos. Novos contribuidores que desejam ingressar em projetos de software livre, por sua vez, costumam enfrentar um conjunto de dificuldades ao tentar contribuir voluntariamente com cada um dos projetos. Visando solucionar tais dificuldades, trabalhos relacionados buscam contribuir com a entrada de novatos em projetos de software livre explorando a receptividade dos projetos. A receptividade de um projeto, neste contexto, pode estar relacionada ao número de novos contribuidores que o projeto retém ao longo de sua história. Quanto mais receptivos os projetos forem, provavelmente maiores serão as chances dos mesmos adquirirem novos contribuidores. Com a finalidade de contribuir com entrada de novatos em projetos de software livre, definiu-se como objetivo desta pesquisa a realização de um estudo exploratório, para investigar a receptividade de novatos em projetos de software livre, bem como a efetividade de indicadores de receptividade. A partir de um conjunto de projetos hospedados na plataforma *GitHub*, foi realizada uma análise de indicadores de receptividade em software livre. Indicadores de receptividade podem ser definidos como um conjunto de indícios capazes de dimensionar o quão receptivos são projetos em software livre. A análise aconteceu por meio de uma relação entre indicadores de receptividade e a média de novos contribuidores semanais por projeto, definida como variável dependente. Os resultados são apresentados comparando projetos com maior e menor receptividade, agrupados entre categorias como domínio, linguagem, idade e tipo de proprietário. Para definir o tamanho da receptividade de um projeto, a média de novos contribuidores em um intervalo de semanas foi calculada para todos os projetos, e estes foram divididos entre acima, e abaixo da média. Encontramos nesta pesquisa que projetos acima da média se apresentam em menor proporção do que os projetos com menor receptividade, e que existem correlações entre a receptividade de um projeto e algumas das características que ele possui. Para os indicadores de receptividade, descobrimos que quando relacionados com base na média de novos contribuidores semanais, as distribuições de projetos abaixo e acima da média possuem alto tamanho de efeito.
Palavras-chaves: Software Livre; Indicadores; Receptividade; Novatos.

Abstract

Dias, Luiz Felipe Fronchetti. Receptivity Analysis in Open Software Projects. 2017. 41. f. Monograph (Undergradute Program in Computer Science), Federal University of Technology – Paraná. Campo Mourão, PR, Brazil, 2017.

Open source projects require voluntary contributions to remain active. New contributors who wish to join an open source project usually face a set of barriers when trying to contribute voluntarily with projects. In order to solve these barriers, a set of related works seek to contribute with newcomers in open source projects by exploring projects receptivity. The receptivity of a project in this context may be related to the number of new contributors contributing to the project. The more receptive the projects are, the more likely they are to receive new contributors. In order to contribute to the onboarding of newcomers into open source projects, the objective of this research was to carry out an exploratory study to investigate the receptivity of newcomers, as well as the effectiveness of receptivity indicators. From a set of projects hosted on the *GitHub* platform, an analysis of receptivity indicators in open source projects was performed. Receptivity indicators can be defined as a set of indicators capable of measuring how receptive are open source projects. The analysis was made through a relationship between receptivity indicators and the average of new weekly contributors per project, defined as the dependent variable. The results are presented comparing projects with greater and less receptivity, grouped among categories such as domain, language, age and owner. To define the receptivity of a project, the average of new contributors over a range of weeks was calculated for all projects, and these were divided between above and below this average. We found that above-average projects are in less proportion than projects with less receptivity, and the existence of correlations between the receptivity of a project and the indicators that it has is significant. For the receptivity indicators, we found that when related to the average of new contributors, above-average projects have a significant effect size compared to the projects below.
Keywords: Open source. Indicators. Receptivity. Newcomers.

Lista de figuras

Lista de tabelas

Sumário

Capítulo 1

Introdução

Software livre introduziu uma nova perspectiva sobre como programas de computador são desenvolvidos (HIPPEL, 2001). Com o auxílio de contribuidores em uma comunidade disposta na internet, contribuições voluntárias são responsáveis pelo desenvolvimento de diversos programas em software livre (WU; LIN, 2001). O código fonte dos programas, disponível publicamente na internet, é construído por contribuidores da comunidade e por programadores externos a ela, que contribuem para o desenvolvimento de cada um dos projetos (CROWSTON; HOWISON, 2005). Essas comunidades para se manterem ativas, recebem periodicamente novos contribuidores com diferentes motivações para contribuir com os projetos.

Entretanto, é comum que, ao tentar ingressar em uma comunidade de software livre, novos contribuidores venham a enfrentar barreiras que dificultem o processo de entrada nos projetos, as quais, por consequência, acabam por atrapalhar a submissão de novas contribuições (STEINMACHER et al., 2014b). De acordo com Qureshi e Fang (2011), o sucesso das comunidades de software livre está intrinsecamente relacionado a participação voluntária de um número significativo de novos desenvolvedores, e é vital para a sustentabilidade de uma comunidade manter tais desenvolvedores no núcleo do projeto.

Pesquisas relacionadas exploram a receptividade de projetos em software livre a partir de diferentes perspectivas, a fim de solucionar algumas das barreiras enfrentadas por novatos. Steinmacher et al. (2015) apresentam um conjunto de barreiras sociais que novos contribuidores enfrentam no processo de entrada em projetos de software livre, e buscam avaliar entre as descobertas como estas barreiras afetam a entrada de novos desenvolvedores. Jensen et al. (2011) apresentam uma análise das primeiras interações entre novos desenvolvedores e contribuidores presentes nas comunidades de software livre, a fim de compreender de que maneira estas relações ocorrem e os desafios enfrentados por novos desenvolvedores ao ingressar em uma comunidade de software livre.

Um conjunto de trabalhos desenvolvidos pela própria comunidade de desenvolvedores em projetos de software livre também busca explorar e solucionar os problemas de receptividade que novatos enfrentam. *Up For Grabs*[1], por exemplo, consiste em um website utilizado para divulgação de pequenas tarefas em projetos de software livre que possam ser desenvolvidas por novos contribuidores, de modo a proporcionar a entrada de novatos em comunidades de software livre por intermédio de pequenas contribuições. Estas pesquisas e trabalhos relacionados buscam de alguma forma viabilizar o encontro entre os projetos de software livre e os novos contribuidores. Entretanto, nenhum dos trabalhos apresentados busca explorar a receptividade dos projetos em relação aos novos contribuidores, de forma a estabelecer indicadores que possam avaliar a receptividade de projetos de software livre.

Nesse sentido, esta pesquisa concentra-se em avaliar a receptividade de projetos em software livre, de modo a compreender e evidenciar indicadores que indiquem projetos mais propensos a serem receptivos a novos contribuidores. Para que fosse possível realizar tal proposta, um conjunto de indicadores de receptividade foram definidos com base na literatura, tais como orientação ao novo contribuidor (existência do arquivo *README.md*, existência do arquivo *CONTRIBUTING.md*), caminhos por onde começar (existência do recurso *Project Board*, existência do recurso Gerenciador de tarefas) e estatísticas do projeto (Média de contribuições por mês, média de estrelas por mês). A receptividade dos projetos foi obtida a partir da média de entrada de novos contribuidores em cada projeto, para um intervalo definido de semanas. Os projetos com maior média de novos contribuidores foram considerados mais propensos a serem receptivos. Na busca por justificar as razões pelas quais projetos tendem a receber o número de novos contribuidores que recebem, encontramos que indicadores de receptividade numéricos, como a média de contribuições por mês, estão fortemente relacionados a entrada de novos contribuidores em um projeto, ao passo que indicadores categóricos, como a existência do arquivo *README.md*, não necessariamente possuem significativa relação com a retenção de contribuidores. Organizamos os projetos por um conjunto de categorias as quais eles se enquadram e descobrimos que, entre as categorias, a linguagem predominante no repositório de código de um projeto pode ser considerada a mais impactante em sua retenção.

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

Capítulo 2

Referencial Teórico

Neste capítulo, serão apresentados trabalhos de outros autores que fundamentam e se relacionam com o contexto desta pesquisa. O objetivo, ao final deste capítulo, é introduzir o leitor quanto ao cenário de software livre, suas comunidades e tecnologias, as barreiras enfrentadas por novatos ao tentar ingressar neste meio e a importância da receptividade em projetos de software livre para melhor receber estes novatos.

2.1. Software livre: Visão geral

Nas últimas décadas, o movimento de software livre cresceu significativamente entre os setores do comércio e da tecnologia, tornando-se responsável por fomentar conferências, investigações acadêmicas e empreendimentos(KELTY, 2001). De acordo com Stallman (2002), software livre é dado como o software que respeita a liberdade e o senso de comunidade dos usuários. Diferente do software proprietário, que restringe a liberdade de compartilhamento e acesso ao código fonte, o software livre permite ser executado, modificado, estudado e redistribuído.

O processo de desenvolvimento do software livre ocorre de maneira colaborativa (WU; LIN, 2001). O código fonte do projeto é disponibilizado publicamente por meio da internet, meio pelo qual desenvolvedores acessam o projeto, adquirem uma cópia do código fonte, e implementam extensões ou correções para o software em questão (HERTEL et al., 2003). As alterações elaboradas pelos desenvolvedores são submetidas à avaliação dos responsáveis pelos repositórios de código, são discutidas e, quando convenientes, adicionadas ao código fonte principal. O processo se repete entre os demais desenvolvedores, de maneira contínua, e a cada progresso da evolução do software desenvolvido.

O modo como projetos de software livre se estruturam socialmente pode variar, e por este motivo a estrutura e sua organização são amplamente estudadas pela literatura. De acordo com Crowston e Howison (2005), a estrutura de uma comunidade de software livre

pode ser definida entre camadas, como apresenta a Figura 2.1. Na mais interna camada estão os principais desenvolvedores, conhecidos como os desenvolvedores do núcleo, aqueles que contribuem com a maior parte do código e supervisionam o projeto, bem como sua evolução. Uma camada acima, ao meio da estrutura, estão os co-desenvolvedores, responsáveis por enviar contribuições (*patches*) de atualizações e correções ao projeto. Na última camada, próxima ao exterior da estrutura, estão os usuários ativos, que não contribuem com código, mas que fornecem casos de uso e relatórios de erros (*bugs*) ao projeto. Por fim, nas extremidades, com um limite praticamente incognoscível, estão os usuários passivos do software, que não contribuem diretamente com o projeto, mas usufruem de seus recursos.

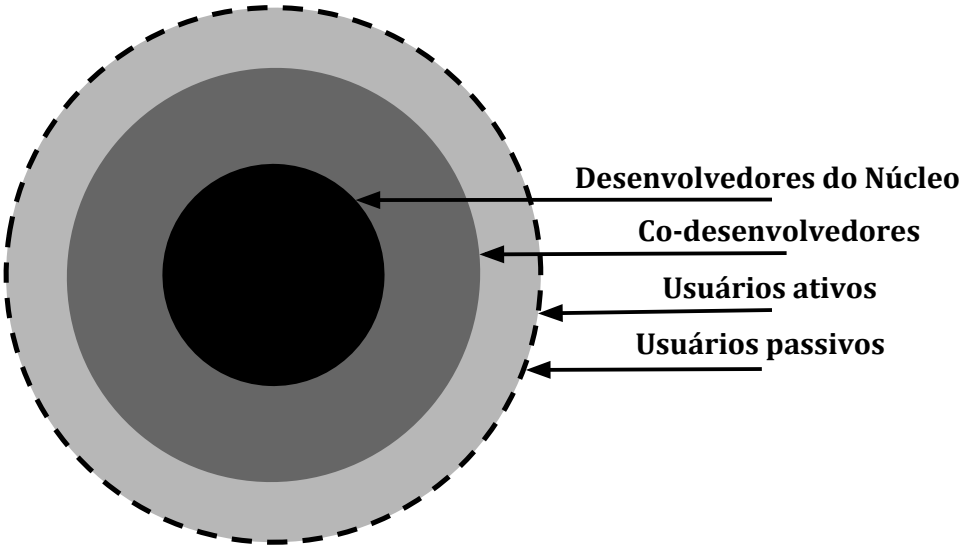


Figura 2.1. Estrutura em camadas de uma comunidade de software livre.
Fonte: https://worldlibraries.dom.edu/index.php/worldlib/article/view/90/27

Na grande maioria dos casos, os projetos de software livre são desenvolvidos por programadores e usuários reunidos em uma comunidade geograficamente distribuída. A participação é voluntária e os contribuidores não recebem uma remuneração direta por seu trabalho (MADEY et al., 2002). De acordo com Hars e Ou (2001), entre as motivações que levam os membros da comunidade a contribuirem voluntariamente com software livre, encontram-se fatores internos e externos a percepção do próprio contribuidor, tais como marketing pessoal, capital humano e determinação própria. Apesar das motivações estarem muitas vezes relacionadas a fatores externos, todas estão comumente relacionadas as características e objetivos do próprio contribuidor.

Assim como em qualquer organização tecnológica, os projetos de software livre dependem de um conjunto de tecnologias para serem desenvolvidos. Tais tecnologias não só

auxiliam os projetos, como fornecem um conjunto de bases de dados para pesquisas acadêmicas. Na próxima seção, serão discutidas algumas das tecnologias utilizadas por projetos de software livre, como o *GitHub*, plataforma de hospedagem de códigos e desenvolvimento colaborativo.

2.2. Tecnologias em software livre

O desenvolvimento de software livre ocorre de maneira voluntária e conta com a dedicação de programadores da comunidade para que cada software possa ser desenvolvido. Devido a necessidade de se organizar o processo de desenvolvimento do software, um conjunto de ferramentas e plataformas surgiram para administrar tais contribuições e, consequentemente, contribuir com o processo de desenvolvimento dos programas. Algumas destas ferramentas não só contribuem com o processo de desenvolvimento, bem como são capazes de armazenar informações relacionadas ao projeto. Estas informações podem se tornar significativas em investigações acadêmicas, motivo pelo qual algumas destas ferramentas serão apresentadas a seguir. Para esta pesquisa, a plataforma de hospedagem de código dos projetos e o ambiente de codificação serão analisados.

As plataformas de hospedagem de código são responsáveis por armazenar todos os arquivos de um determinado projeto em uma estrutura denominada repositório de código. Atualmente, é comum que estas plataformas se encontrem junto aos ambientes de codificação, meio pelo qual desenvolvedores realizam, discutem e submetem contribuições. Os ambientes de codificação não só envolvem as contribuições de código, bem como a administração das contribuições externas, e apresentação de relatórios e erros dispostos pela comunidade. Devido a sua notória presença em pesquisas recentes (por exemplo, (PINTO et al., 2016), (MOURA et al., 2015), (TSAY et al., 2014)), a plataforma de hospedagem de código *GitHub*, descrita abaixo, será utilizada como fonte de dados nesta pesquisa.

2.2.1. GitHub

GitHub é um website de hospedagem de código colaborativo para projetos de software livre (gratuitos) e proprietários (pagos), construído sob o sistema de controle de versão *git*[1]. Com mais de 19 milhões de repositórios de código hospedados[2], o *GitHub* pode ser considerado uma entre as mais importantes fontes de projetos de software na internet, além de se enquadrar entre os maiores websites de hospedagem de códigos do mundo.

A plataforma *GitHub* é uma das plataformas responsáveis por popularizar ao âmbito do desenvolvimento de software colaborativo o modelo "*forks e pull-requests*", onde contribuidores em projetos de software livre são capazes de propor mudanças ao código sem necessariamente terem direitos sobre o repositório principal do projeto (YU et al., 2016).

[1] https://git-scm.com/
[2] https://octoverse.github.com/

Neste modelo, os contribuidores criam uma cópia (*fork*) do repositório principal por meio da plataforma e trabalham nela de maneira independente, com total controle e liberdade. Quando necessário, requisitam que suas mudanças sejam integradas ao repositório principal através de um *pull-request*, nome dado a este tipo de requisição. Desta maneira, o modelo permite que projetos recebam contribuições externas a comunidade sem necessariamente conceder acesso e controle prévio ao repositório principal, possibilitando, inclusive, espaço para discussões sobre as contribuições de terceiros, antes mesmo delas serem aceitas.

O *GitHub* também possuí, para cada projeto, seu próprio sistema de controle de erros (*bugs*) e questões (*issues*) a serem debatidas pela comunidade que os desenvolve. Neste sistema, conhecido como caça tarefas (*issue tracker*), é possível não só relatar problemas e discutir questões referentes ao software desenvolvido, mas também abrir discussões entre os desenvolvedores e usuários de cada comunidade.

Outros dois recursos importantes apresentados pelo *GitHub* que o diferencia em relação aos demais ambientes de codificação são os sistemas de *Wiki* e *Project Board*. No sistema de *Wiki*, como mostra a Figura 2.2, os desenvolvedores do projeto descrevem detalhadamente as características dos códigos desenvolvidos dentro do repositório, as etapas de instalação do projeto e informações que usualmente são pertinentes tanto para os usuários bem como para os desenvolvedores. A utilização da *Wiki* dentro do próprio ambiente de codificação pode facilitar a escrita e a procura de documentações, visto que a necessidade de um website específico para hospedagem destes documentos passa a ser opcional.

O *Project Board*, por sua vez, é um sistema integrado a cada projeto hospedado no *GitHub*, que viabiliza a organização e a priorização de trabalhos entre os desenvolvedores do repositório. O *Project Board* permite aos desenvolvedores de um projeto organizar tarefas e questões (*issues*) a serem resolvidas, bem como requisições (*pull-requests*) a serem analisadas, além de possibilitar a criação de anotações a serem discutidas pela equipe. Os sistemas de caça-tarefas (*issue tracker*), *Wiki* e *Project Board* são opcionais para todos os projetos criados dentro da plataforma, e cabe aos proprietários do projeto decidirem o uso dos mesmos.

Apesar de sua notoriedade, o *GitHub* não é o único ambiente de codificação utilizado por desenvolvedores de software. Existem diversas opções, como o *CodePlex*, específico para tecnologias *Microsoft*, e o *BitBucket*, que permite trabalhar com diferentes sistemas de versionamento de código. No entanto, tem sido comum a migração de várias comunidades de software para o *GitHub*, em particular devido a sua notória popularidade, bem como suas funcionalidades sociais que facilitam a criação e gerenciamento de times colaborativos (DABBISH et al., 2012; GOUSIOS et al., 2014).

É perceptível que projetos de software livre estão cercados por um conjunto de ferramentas capazes de contribuir com o processo de desenvolvimento de software, de modo com que não só desenvolvedores da comunidade, mas também contribuidores externos à ela, estejam aptos a realizarem contribuições ao projeto. Entretanto, apesar da existência destas

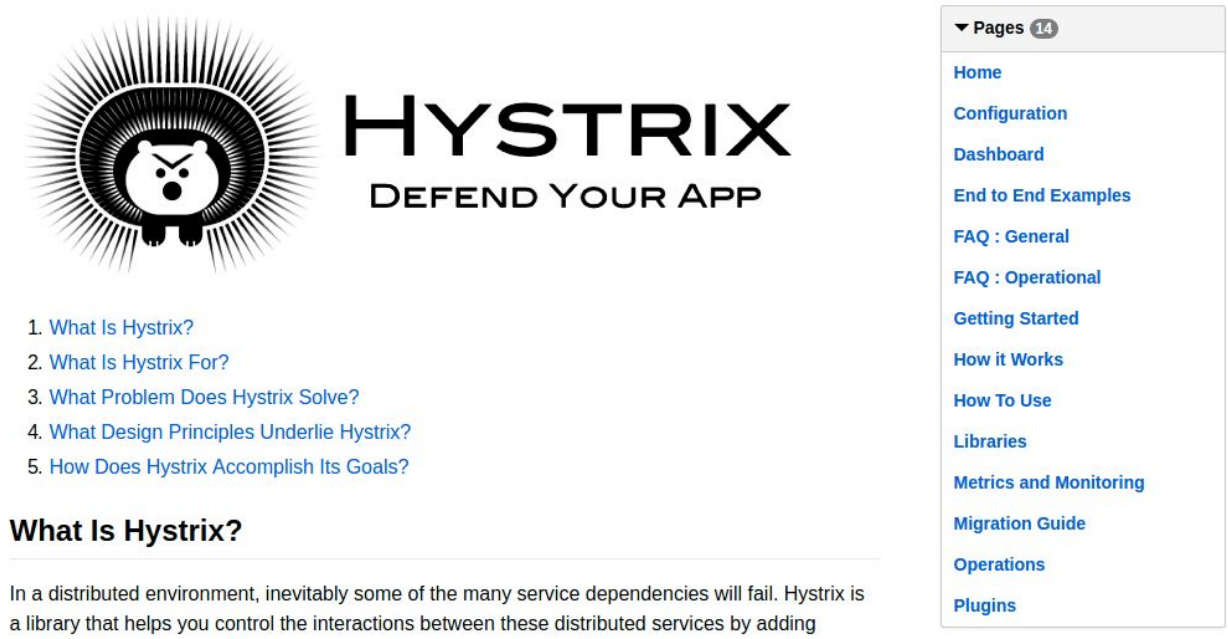


Figura 2.2. Utilização do sistema *Wiki* no *Hystrix*, biblioteca desenvolvida pela *Netflix*.

tecnologias, aqueles que desejam ingressar em uma comunidade de software livre usualmente enfrentam um conjunto de barreiras ao tentar submeter uma primeira contribuição ao projeto de software (STEINMACHER et al., 2015). Estas barreiras não só dificultam o processo de contribuição do novo contribuidor, como prejudicam o processo de desenvolvimento do software. Uma visão geral sobre estas barreiras pode ser observada na seção a seguir.

2.3. Barreiras enfrentadas por novatos em projetos de software livre

As contribuições em software livre ocorrem de maneira voluntária e algumas evidências na literatura descrevem a necessidade contínua de novos contribuidores em projetos de software livre. Neste contexto, um novo contribuidor não é necessariamente definido como aquele que não domina o contexto de software livre, do desenvolvimento de software ou das linguagens de programação, mas sim aquele que deseja ingressar pela primeira vez em uma nova comunidade, em um projeto de software livre.

É comum que nesta forma de manifestação, novatos venham a enfrentar barreiras ao tentar contribuir com projetos de software livre (HANNEBAUER et al., 2014). Estas barreiras são definidas como um conjunto de dificuldades, empecilhos ou deficiências no processo de desenvolvimento, que impossibilitam o contribuidor em questão a realizar sua primeira contribuição (KROGH et al., 2003). Steinmacher et al. (2015) identificaram em sua pesquisa um conjunto de barreiras enfrentadas por novatos ao tentar contribuir com software livre. As barreiras foram organizadas em categorias, relacionadas a causas referentes aos

projetos de software livre, bem como aos próprios novatos. Problemas de documentação, diferenças culturais e obstáculos técnicos estão entre as categorias de barreiras relacionadas aos projetos. Entretanto, não só o projeto, mas também o próprio novato tem participação na construção destas barreiras, como a falta de comunicação do novato para com os membros do projeto e a falta de conhecimento prévio em relação as linguagens de programação, processos e diretrizes utilizados no projeto a ser desenvolvido.

Outros estudos avaliam possíveis barreiras em demais contextos do desenvolvimento de software livre. Jensen et al. (2011), por exemplo, ao analisar listas de e-mail de projetos de software livre, encontraram que novatos que recebem respostas em um curto espaço de tempo tendem a ter uma participação no projeto. Em contrapartida, quando novos contribuidores recebem respostas grosseiras, estes possivelmente se sentem desmotivados a contribuir com o projeto.

Estas e outras barreiras acabam por acarretar na desistência do novato em contribuir com software livre. Um conjunto de desistências, por sua vez, enfraquece o processo de desenvolvimento do software, que necessita destes novos contribuidores. Para que seja possível solucionar as barreiras existentes, as quais usualmente impedem que novos contribuidores realizem contribuições, é necessário que um conjunto de estudos e possíveis tecnologias sejam criados a fim de auxiliar os novatos bem como os projetos que necessitam destes contribuidores. Na próxima seção, serão discutidos trabalhos desenvolvidos pela comunidade que buscam suprir de diferentes maneiras algumas das barreiras enfrentadas por novos contribuidores.

2.4. Ferramentas para apoiar novatos em projetos de software livre

Uma diversidade de projetos disponíveis na internet buscam auxiliar novos contribuidores no processo de contribuir com software livre. Nesta seção, serão apresentados um conjunto de ferramentas relacionadas a proposta desta pesquisa, que de alguma forma buscam solucionar as barreiras existentes entre novatos e projetos de software livre, bem como, promover a receptividade em projetos.

2.4.1. *Up For Grabs*

Up For Grabs[3] é um website que tem como objetivo auxiliar novatos por meio de tarefas a contribuir com software livre. Os novatos acessam o website e a eles é apresentado uma série de projetos que contenham tarefas nos sistemas de caça tarefas (*issue tracker*) específicas para novos contribuidores. Estas tarefas são identificadas por rótulos que condizem com atividades que possam ser desenvolvidas por novos contribuidores, os rótulos são cadastrados

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

pelos contribuidores responsáveis pelo projeto. A ideia é que, ao tentar contribuir com um projeto, o novato saiba por quais tarefas começar, sem necessariamente propor uma nova funcionalidade, ou identificar erros a serem solucionados no software. A Figura 2.3 mostra um exemplo de projeto cadastrado no website.

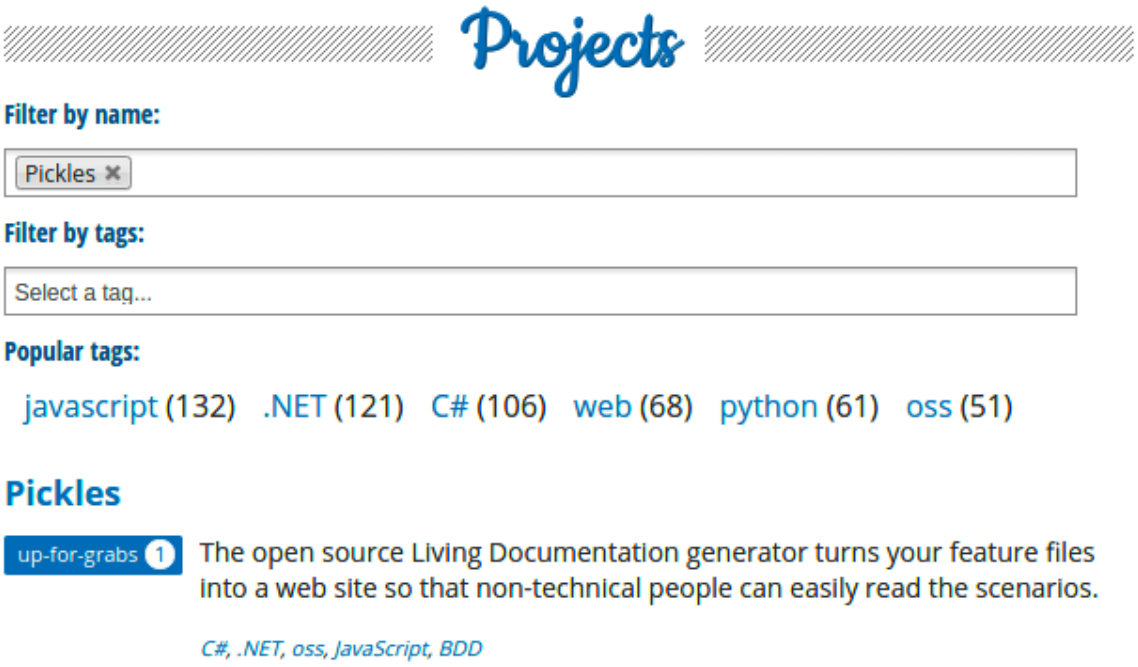


Figura 2.3. Exemplo de projeto cadastrado no *Up For Grabs*. *Pickles* é o nome do projeto, e *up-for-grabs* é o rótulo das tarefas destinadas aos novatos.

Os projetos de software livre, que desejam entrar na lista do website, submetem um arquivo com a descrição do projeto e o identificador referente ao rótulo usado nas tarefas dedicadas aos novatos. O website é simples e objetivo, tendo como principais recursos a lista de projetos cadastrados e um sistema de filtragem de projetos por rótulo e nome.

2.4.2. *First Timers Only*

First Timers Only[4], projeto relacionado ao *Up For Grabs*, é um website encarregado de estabelecer diretrizes para novatos que desejam contribuir com software livre. Nesta página é possível encontrar uma diversidade de dicas e tutoriais apresentados para quem deseja contribuir com software livre pela primeira vez. Entre as páginas apresentadas pelo website, está o próprio *Up For Grabs*, assim como uma lista de recomendações com tutoriais, documentários e códigos de conduta destinados à novos contribuidores.

[4] http://www.firsttimersonly.com/

Além de fornecer caminhos para que novatos ingressem em software livre, a página também apresenta dicas para que projetos tornem mais receptiva a entrada de novos contribuidores. A principal recomendação do website quanto aos repositórios se concentra em encorajar que projetos de software livre utilizem rótulos em tarefas nos sistemas de caça tarefas que sejam dedicadas aos novatos. Tais tarefas, segundo a página, devem condizer com a capacidade de compreendimento do novo contribuidor em relação ao projeto desenvolvido.

2.4.3. *Your First Pull Request*

Na rede social *Twitter*[5], o projeto *Your First Pull Request*[6] divulga semanalmente através de sua conta, tarefas destinadas a iniciantes que desejam ingressar em projetos de software livre, e possibilita, por meio de mensagens publicadas na rede (*tweets*), que projetos divulguem sua busca por novos contribuidores. A finalidade deste trabalho é tornar comunidades de software livre mais acessíveis a novos contribuidores por meio de uma divulgação abrangente, ajudando os mantenedores das comunidades a diminuir as barreiras que dificultam a entrada de novatos. Para que um projeto seja divulgado pelo *Your First Pull Request*, basta que o divulgador o mencione na mensagem a ser divulgada, como mostra a Figura 2.4.

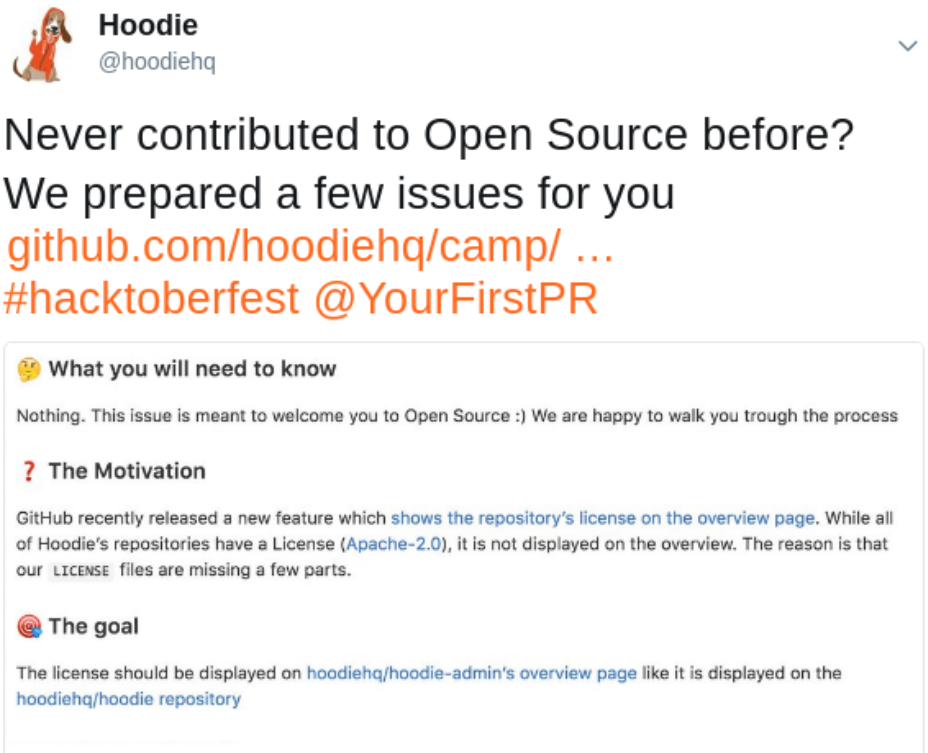


Figura 2.4. Exemplo de mensagem divulgada pelo *Your First Pull Request.* Mensagem publicada pelo projeto *Hoodie*, um serviço para aplicações web.

[5] https://twitter.com
[6] https://twitter.com/yourfirstpr

Apesar dos projetos desenvolvidos contribuírem com a entrada de novos contribuidores, nenhum deles explora a receptividade de projetos em software livre. Os trabalhos apresentados compreendem características intrinsecamente relacionadas aos novatos, mas não necessariamente aos projetos em si. Em nossa pesquisa, uma abordagem focada especificamente na receptividade dos projetos de software livre será apresentada. Para tal, apresentamos a seguir uma visão geral sobre receptividade em software livre, com ênfase nas diferenças entre os estudos apresentados e o que foi trabalhado nesta pesquisa.

2.5. Receptividade de novos contribuidores em projetos de software livre

Um dos grandes desafios enfrentados por comunidades de software livre encontra-se em possibilitar que novos contribuidores ingressem a comunidade e, de fato, realizem contribuições (GOUSIOS et al., 2016). Visto que, para um projeto de software livre permanecer ativo contribuições devem ser realizadas periodicamente, o ingresso de novos contribuidores passa a ser esperado, e a recepção destes novatos deve ser levada em consideração.

A receptividade de um projeto pode ser relacionada ao número de contribuidores que o projeto retém. Quanto mais receptivos os projetos forem, provavelmente maiores serão as chances dos mesmos adquirirem novos contribuidores (YE; KISHIDA, 2003). Steinmacher et al. (2014a) exploram a receptividade de novatos propondo um modelo de entrada de novos contribuidores. O modelo visa suprir algumas das barreiras anteriormente evidenciadas (STEINMACHER et al., 2013a), e está fundamentado em quatro diferentes forças que influenciam o progresso de um novato em software livre, que são: motivação, atratividade, retenção e iniciativa para fatores presentes no processo de entrada. Quando projetos de software livre atendem as forças requiridas por novos contribuidores, usualmente o processo de recepção pode vir a se tornar mais sólido.

Em uma outra perspectiva, Canfora et al. (2012) propõem uma abordagem de identificação de mentores em projetos de software livre denominada Yoda (*Young and newcOmer Developer Assistant*). Através da mineração de dados em listas de e-mail e sistemas de controle de versão, o sistema consegue identificar potenciais membros do projeto que podem auxiliar novos desenvolvedores a ingressar na comunidade e a realizar contribuições, construindo uma relação de receptividade entre membros do projeto e novos contribuidores. Entre as descobertas desta pesquisa, encontram-se evidências que não só os desenvolvedores que mais contribuem com o repositório de código são indicados como os principais mentores, mas outras categorias de contribuidores também podem ser indicadas para processos de orientação, destacando a necessidade de uma comunidade colaborativa.

A receptividade na grande maioria dos estudos relatados nesta seção está relacionada a uma abordagem do novo contribuidor para com o projeto. A perspectiva destas pesquisas

geralmente está apoiada em processos que novos contribuidores enfrentam, como instalar as dependências do repositório ou encontrar alguém que os auxilie a contribuir. Entretanto, existem outros meios para se tratar de receptividade em software livre, como por exemplo, encontrar evidências a partir dos próprios projetos que demonstrem processos de relação com novos desenvolvedores, sem necessariamente utilizar como ênfase da pesquisa o próprio novato. Em nossa pesquisa, uma abordagem focada especificamente nos projetos será apresentada, de modo a compreender algumas das características que projetos de software livre apresentam que, de certa forma, possam justificar o número de novos contribuidores que estes projetos recebem.

Acreditamos que estas características possam ser apresentadas através de indícios encontrados nos projetos de software livre, como em arquivos de registro, históricos, estatísticas e entre outros dados relacionados ao projeto. Definimos estes indícios como indicadores de receptividade, sendo um indicador um nome atribuído a alguma evidência do projeto que, de certa forma, justifique sua receptividade. Outros estudos já utilizaram o conceito de indicador para investigar alguma evidência no contexto de software livre, como por exemplo Borges et al. (2016a), que investigaram padrões de popularidade em projetos hospedados no *GitHub* através de um conjunto de informações do próprio projeto, e Coelho e Valente (2017), que evidenciaram alguns dos motivos pelos quais projetos de software livre fracassam, utilizando no método arquivos e recursos do projeto. Todavia, nenhum destes estudos investigou a receptividade dos projetos em comunidades de software livre.

2.6. Considerações finais

Os trabalhos apresentados neste capítulo buscam compreender aspectos relacionados a projetos de software livre e, de certo modo, viabilizar o estudo de novos contribuidores neste contexto. Quanto as ferramentas apresentadas, a divulgação dos projetos e a ajuda fornecida a novos contribuidores podem ser destacados como os grandes diferenciais que as tornam popularmente conhecidas pelo público que usufrui de seus recursos. Tanto os trabalhos como as ferramentas auxiliam o processo de entendimento das circunstâncias relacionadas a esta pesquisa, mas, nenhum dos trabalhos apresentados busca explorar a receptividade dos projetos de software livre quanto a retenção de novos contribuidores, de forma a estabelecer indicadores que possam avaliar a receptividade dos projetos.

Pelas razões evidenciadas acima, apresentamos no capítulo a seguir a metodologia desenvolvida nesta pesquisa, onde exploramos características relacionadas a receptividade de um conjunto de projetos de software livre, em busca de estabelecer potenciais valores em cada projeto que justifiquem o número de contribuidores que estes recebem.

Capítulo 3

Metodologia

Nesta pesquisa, exploramos a receptividade de novos contribuidores em projetos de software livre, de modo a compreender por meio de indicadores de receptividade o que torna mais propensa a recepção de novatos, e discutimos as razões pelas quais projetos tendem a receber novos contribuidores. A seguir são descritos o objetivo, questão de pesquisa e o método de pesquisa executado.

3.1. Objetivo

Com a finalidade de contribuir com entrada de novatos em projetos de software livre, definiu-se como objetivo desta pesquisa a realização de um estudo exploratório, para investigar a receptividade de novatos em projetos de software livre, bem como a efetividade de indicadores de receptividade, de modo que fosse possível favorecer novas contribuições e, por consequência, prover atividade a projetos de código aberto.

Para que fosse possível avaliar a receptividade de novos contribuidores, bem como os indicadores de receptividade, selecionamos um conjunto de projetos de código aberto hospedados na plataforma *GitHub*. Foram coletados para cada um dos projetos, indicadores de receptividade referentes a todo histórico de atividades dos projetos, definidos como um conjunto de indícios capazes de dimensionar o quão receptivos são projetos de software livre. O objetivo ao final desta pesquisa consiste em avaliar quais dos indicadores propostos estão aptos a indicar o que favorece a receptividade de projetos de software livre, e identificar quais indicadores costumam estar presentes em projetos propensos a serem receptivos a novos contribuidores. Para definir a receptividade dos projetos, utilizamos as últimas 14 semanas em que o projeto esteve em atividade antes dos dados serem coletados, de modo que fosse possível entender como os indicadores afetam a recente receptividade de cada projeto estudado.

Objetivo: *Explorar a receptividade de novos contribuidores e a efetividade de um conjunto de indicadores de receptividade em projetos de software livre.*

Considerando que o processo de entrada de novos contribuidores em projetos de software livre é uma atividade fundamental para a manutenção e evolução de software (STEINMACHER et al., 2014a) e dado o objetivo estabelecido para esta pesquisa, foi executada uma análise das relações entre indicadores de receptividade e o número de novos contribuidores que projetos de software livre recebem, de modo a compreender as razões pelas quais projetos de software livre recebem o número de novos contribuidores que recebem, e os motivos pelos quais projetos considerados receptivos tendem a receber mais contribuidores.

3.2. Método

Para alcançar o objetivo de pesquisa, o método de pesquisa a seguir foi utilizado. Com ênfase em um conjunto de indicadores de receptividade, o método é dividido em: definição dos indicadores, seleção dos projetos, coleta dos indicadores e avaliação dos projetos, como mostra a Figura 3.1.

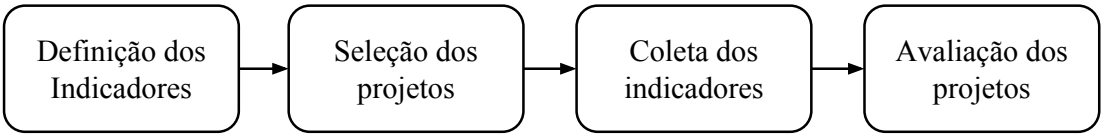


Figura 3.1. Etapas propostas no método de pesquisa

A etapa de definição dos indicadores consistiu na determinação, baseada em trabalhos relacionados, de quais indicadores de receptividade seriam utilizados. A etapa de seleção dos projetos estabeleceu quais projetos serviriam como fonte de coleta para os indicadores definidos. A coleta também foi definida como uma etapa, feita através da *GitHub API* e dos repositórios de código de cada projeto. Por fim, a última etapa consistiu na avaliação dos projetos quanto a sua receptividade, levando em consideração os indicadores coletados na etapa anterior. Nas próximas subseções, apresentaremos uma visão geral de cada uma destas etapas relacionadas ao nosso método.

3.2.1. Indicadores de receptividade

Indicadores de receptividade podem ser definidos como um conjunto de indícios capazes de dimensionar o quão receptivos são projetos de software livre. Para melhor apresentação, o conjunto de indicadores foi definido em três categorias distintas, cada qual responsável por abranger barreiras apresentadas por Steinmacher et al. (2015), potencialmente presentes em

cada projeto. Os indicadores de receptividade e suas respectivas categorias definidas nesta pesquisa são apresentados na Tabela 3.1.

Tabela 3.1. Indicadores de receptividade e suas respectivas categorias

Categoria	**Indicadores**
Orientação ao novo contribuidor	(1) Existência do arquivo *README.md* (2) Existência do arquivo *CONTRIBUTING.md* (3) Existência do recurso *Wiki*
Caminhos por onde começar	(4) Existência do recurso *Project Board* (5) Existência do recurso de Gerenciador de tarefas (*Issue Tracker*)
Estatísticas do projeto	(6) Média do número de requisições abertas por mês (*Opened Pull-requests*) (7) Média do número de requisições fechadas por mês (*Closed Pull-requests*) (8) Média do número de requisições aceitas por mês (*Merged Pull-requests*) (9) Média do número de contribuições por mês (*Commits*) (10) Média do número de estrelas por mês (*Stars*) (11) Média do número de cópias por mês (*Forks*)

Orientação ao novo contribuidor O processo de orientação está relacionado a todo ou qualquer tipo de indicação estabelecida pelo projeto com o objetivo de esclarecer detalhes técnicos e estruturais do que é desenvolvido e que, por consequência, venha a facilitar o entendimento do novato quanto a estrutura e metodologia de desenvolvimento de códigos do repositório. A escolha dos indicadores desta categoria é baseada nos trabalhos de Lethbridge et al. (2003), que definem documentação de arquiteturas e demais informações abstratas como necessárias, ou pelo menos fornecedoras de orientação histórica útil para mantenedores do projeto, e Forward e Lethbridge (2002), que evidenciam a documentação de estruturas e códigos, e a utilização de comentários como as mais importantes para manutenção de um sistema. Acreditamos, baseados nestes trabalhos, que os arquivos e sistema apresentados nesta categoria tenham relação com potenciais orientações aos novatos.

Caminhos por onde começar. Estudos apontam que novatos necessitam de atenções especiais ao entrar em contato com software livre (CAPILUPPI; MICHLMAYR, 2007; BEN et al., 2013). A orientação e recepção destes novatos se tornam essenciais para que eles não abandonem o processo de contribuição (STEINMACHER et al., 2013b). Entre os pontos que podem tornar um projeto receptivo a novas contribuições, de acordo com os estudos relacionados, estão disponibilizar acesso ao modo como as contribuições ocorrem, e permitir que novatos encontrem questões a serem resolvidas que condizem com a sua atual percepção em relação a complexidade do projeto.

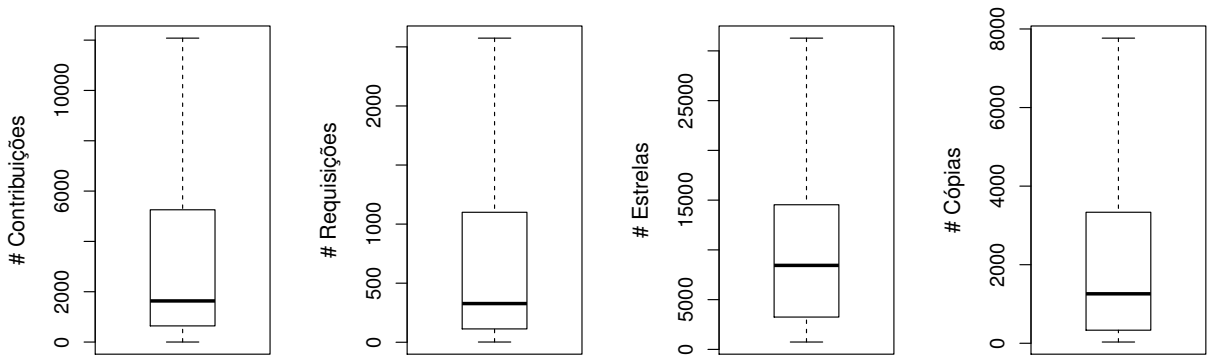
Estatísticas do projeto. Esta categoria busca estabelecer medidas quantitativas relacionadas as características do projeto, como o número de contribuições de um projeto e a média de requisições externas aceitas por ele. A ideia neste caso, é levantar medidas que possam categorizar a receptividade de um projeto a partir do seu histórico de atividades,

proposta baseada nos estudos de Borges et al. (2016b), que ao avaliar popularidade de repositórios de código no *GitHub*, correlacionou o número de estrelas de um repositório (variável dependente) com cópias, contribuições, contribuidores e idade dos projetos. Outros artigos também investigam indícios quantitativos em outros contextos do desenvolvimento de software, Tian et al. (2015), por exemplo, investigam em sua pesquisa 28 fatores que buscam explicar porque aplicativos com alta reputação no sistema *Android* se diferenciam de aplicativos com baixa reputação.

3.2.2. Seleção dos projetos

Para que fosse possível avaliar os identificadores e a recente receptividade proposta nesta pesquisa, 450 projetos de software livre hospedados na plataforma *GitHub* foram selecionados. Para evitar que os projetos selecionados não contivessem contribuições externas, estivessem inativos ou que não fossem de fato projetos de software, a seleção dos projetos seguiu a mesma metodologia proposta por Ray et al. (2014), onde os projetos escolhidos foram selecionados em ordem decrescente de popularidade (de acordo com o número de estrelas[1]) e divididos igualmente entre as 15 principais linguagens de programação: `C`, `Clojure`, `CoffeeScript`, `Erlang`, `Go`, `Haskell`, `Java`, `JavaScript`, `Scala`, `Objective-C`, `Perl`, `PHP`, `Python`, `Ruby` e `TypeScript`.

Os projetos selecionados compuseram uma grande variedade em número de contribuições (média = 7618, mediana = 1634, desvio padrão = 34723), requisições (média = 1433, mediana = 327, desvio padrão = 3637), estrelas (média = 12065, mediana = 8442, desvio padrão = 17981) e cópias por repositório (média = 2676, mediana = 1259, desvio padrão = 4178), o que demonstra que estes podem ser considerados ativos em suas comunidades, como mostra a Figura 3.6.



(a) Contribuições (b) Requisições (c) Estrelas (d) Cópias

Figura 3.6. Distribuição de contribuições, requisições, estrelas e cópias entre os projetos selecionados para esta pesquisa

[1] https://help.github.com/articles/about-stars/

Quanto aos contribuidores por projeto, os números obtidos também são diversos. De acordo com as linguagens de programação utilizadas, como mostra a Tabela 3.2, encontramos que os projetos com maior número de contribuidores são usualmente escritos em `Ruby`, com uma média de 921 contribuidores por projeto (desvio padrão = 1486), seguidos de projetos escritos em `C` com uma média de 818 contribuidores (desvio padrão = 3129), e logo após projetos escritos em `JavaScript` que contam com uma média de 514 contribuidores (desvio padrão = 504). A menor valor em termos de contribuidores por projeto corresponde ao `Perl`, com uma média de 65 contribuidores (desvio padrão = 147). A idade dos projetos selecionados também varia significativamente, entre menos de um ano até 9 anos de idade.

Tabela 3.2. Distribuição de contribuidores nos projetos por linguagem de programação

Linguagem	**Min**	**Max**	**Média**	**Mediana**	**Desvio padrão**
C	1	17262	818	138	3129
Clojure	10	369	72	61	70
CoffeeScript	4	774	104	37	188
Erlang	8	510	82	6	91
Go	28	1891	402	199	470
Haskell	5	480	98	53	107
Java	19	997	166	95	201
JavaScript	47	1780	514	337	504
Scala	7	1543	195	97	306
Objective-C	9	341	87	60	75
Perl	3	814	65	34	147
PHP	36	1864	366	226	449
Python	1	3491	435	278	671
Ruby	66	6659	921	407	1486
TypeScript	10	5429	328	113	972

Estas informações sugerem que o conjunto de projetos selecionados satisfazem as características necessárias para uma análise de projetos de software livre, visto que os dados coletados compreendem uma diversidade de projetos com diferentes tamanhos e características, como os já utilizados em outras pesquisas relacionadas. Vale ressaltar que todos os dados coletados nesta pesquisa, bem como os códigos escritos na execução da metodologia, estão disponíveis em nosso repositório[2].

[2] <https://github.com/fronchetti/TCC-UTFPR>

3.2.3. Coleta dos indicadores

Após selecionados os projetos a serem avaliados, o próximo passo consistiu em realizar a coleta dos indicadores de receptividade. Para esta tarefa, foram utilizados como fonte de dados os repositórios de código de cada projeto, além de registros disponíveis na *GitHub API*[3]. A maneira como os dados foram coletados se diferencia entre as categorias de indicadores.

Para as categorias **Orientação ao novo contribuidor** e **Estatísticas dos projetos**, três indicadores de receptividade foram coletados através dos repositórios de código. A existência dos arquivos *README.MD* e *CONTRIBUTING.MD* foi verificada utilizando um *script* escrito em `Python`, que verifica a existência dos respectivos arquivos nos repositórios. A média do número de contribuições por mês (*commits*) foi verificada explorando os registros do repositório, através de comandos `git log`. Optamos por extrair as contribuições diretamente do repositório ao invés da *GitHub API*, pois a *API* só apresenta contribuições realizadas dentro do próprio *GitHub*, e não necessariamente todas as contribuições do projeto, incluindo aquelas realizadas em outros ambientes de codificação.

Quanto a categoria **Estatísticas do projeto** e aos demais indicadores das categorias citadas anteriormente, os dados foram coletados por intermédio da *GitHub API*, que provê serviços de busca a repositórios públicos e coleta de dados relacionados a eles (por exemplo, número de estrelas, contribuidores e tarefas do projeto). As médias aritméticas foram calculadas, pela soma total dos valores mensais de um respectivo indicador, dividida pelo número total de meses em que o projeto esteve ativo. A **Média do número de requisições aceitas por mês**, por exemplo, é resultado da soma total do número de requisições aceitas mês a mês, dividido pelo número total de meses em que o projeto esteve ativo. A **Média do número de contribuições por mês** segue abordagem similar, sendo o número de contribuições mensalmente anexadas ao repositório, dividido pelo número total de meses de atividade do projeto. As médias mensais foram utilizadas pois estas provém uma perspectiva mais ampla das atividades realizadas em cada projeto, possibilitando uma análise temporal dos dados coletados.

3.2.4. Avaliação dos projetos

O último passo consistiu em avaliar o conjunto de projetos selecionados quanto a receptividade para novatos, utilizando como parâmetros de avaliação os indicadores coletados. Como esta se trata de uma pesquisa exploratória, o processo de avaliação foi realizado em quatro etapas: coleta de dados, visualização da variável dependente (média de novos contribuidores semanais), relação com os indicadores e análise dos projetos quanto a receptividade de novatos.

O objetivo ao final deste conjunto de etapas foi conseguir identificar, para os projetos coletados, relações entre os indicadores de receptividade propostos e o número de novos

[3] https://api.github.com/

contribuidores que estes projetos receberam nas últimas semanas, de modo a compreender as razões pelas quais projetos de software livre recebem o número de novos contribuidores que recebem, e os motivos pelos quais projetos considerados receptivos tendem a receber mais contribuidores.

Variável dependente: Média de novos contribuidores semanalmente distribuídos

Uma variável dependente representa uma grandeza cujo valor depende de como a variável independente é manipulada. Nesta pesquisa, avaliamos se a recente receptividade dos projetos, caracterizada por intermédio da média de novos contribuidores semanais que um projeto recebeu (variável dependente), varia de acordo com cada um dos indicadores de receptividade propostos nesta pesquisa (variáveis independentes).

A quantidade de semanas coletadas para representar a variável dependente foi definida por um intervalo único para todos os projetos, e não necessariamente pelo total de semanas em que cada projeto esteve em atividade. O intervalo teve início na primeira semana em que o projeto mais recente entre os coletados foi criado, até a respectiva semana em que os dados desta pesquisa foram coletados. Ao utilizar um intervalo de semanas, buscamos unificar o intervalo de tempo analisado entre os projetos, e compreender como o histórico de cada projeto, definido pelos indicadores, pode impactar a recente receptividade dos mesmos. Avaliamos, por exemplo, se a média do número de cópias por mês (*forks*) de um projeto, possuí alguma relação com a média de novos contribuidores que este determinado projeto recebeu nas últimas semanas. Para esta pesquisa, o projeto mais recente é o `chromeless`[4], que foi criado em Junho de 2017, totalizando deste modo um intervalo de 14 semanas analisadas para cada projeto.

A variável dependente foi coletada por meio de registros do repositório, com o uso do comando `git log`. Para avaliar se os dados coletados faziam sentido a proposta da pesquisa, gráficos da distribuição de novos contribuidores foram gerados para cada um dos projetos coletados. Apresentamos também correlações pelo método de Spearman para as distribuições de novos contribuidores e os indicadores de receptividade ($p < 0{,}05$), em busca de uma visão mais ampla sobre a recepção de novos contribuidores aceitos em cada projeto. Neste contexto, um contribuidor é considerado novato de acordo com a exata data em que realizou sua primeira contribuição. Contribuições posteriores realizadas pelo mesmo contribuidor foram descartadas para esta variável.

Relação: variável dependente e indicadores de receptividade

O segundo passo desta proposta consistiu em relacionar, para cada um dos projetos, os indicadores propostos na Tabela 3.1. O objetivo ao estabelecer esta relação foi de encontrar

[4] <https://github.com/graphcool/chromeless>

evidências que, para cada um dos indicadores, haveria uma potencial relação com o número de novos contribuidores que um projeto recebeu recentemente. Foi possível, por exemplo, responder se a existência do arquivo *README.md* estava relacionada ao número de novos contribuidores que um projeto recebeu. Para facilitar a visualização das relações, os projetos foram agrupados entre os mais receptivos e os menos receptivos, utilizando a variável dependente desta pesquisa como ordem de seleção.

Como os indicadores se distinguem entre valores numéricos e categóricos, duas abordagens foram utilizadas para o cálculo das relações. Para avaliar as comparações entre valores numéricos, como os indicadores da categoria **Estatísticas do projeto**, foram utilizados os testes estatísticos não-paramétricos Mann-Whitney-Wilcoxon (MWW) e o delta de Cliff (tamanho do efeito) (SIEGEL; JR, 1975). Os testes foram escolhidos porque as distribuições de indicadores numéricos não seguem uma distribuição normal. O teste MWW foi utilizado para verificar se as duas distribuições são diferentes para um $\alpha = 0,05$. O delta de Cliff foi utilizado para dimensionar o tamanho da diferença entre as distribuições. Quanto maior o valor do delta de Cliff obtido, maior era a diferença entre as distribuições apresentadas. Para interpretar os resultados do tamanho da diferença foi utilizada a escala provida por Romano et al. (2006): $delta < 0,147$ (diferença insignificante), $delta < 0,33$ (diferença baixa), $delta < 0,474$ (diferença média), $delta >= 0,474$ (diferença alta).

Para avaliar as comparações entre valores categóricos, tais como a Existência do recurso *Wiki*, o teste exato de Fisher foi utilizado (UPTON, 1992). O teste de Fisher é um teste de significância estatística destinado a avaliar a hipótese de que os efeitos de linha e coluna são independentes em uma tabela de contingência (2×2). O princípio básico do método de Fisher é comparar proporções, ou seja, verificar se existe associação não aleatória entre dois conjuntos de dados, tais como a média de novos contribuidores semanais e os indicadores de receptividade categóricos, sendo esta uma alternativa para testes como Qui Quadrado, utilizado quando o quantidade de valores ultrapassa mil amostras, o que não é o nosso caso. Para avaliar os resultados obtidos no teste de Fisher, nós verificamos se existe significância estatística ($p <$ 0,01), com grau de confiança de 95%, e caso a condição seja satisfeita, rejeitamos a hipótese nula (H0: a variável da linha e a variável de coluna são independentes). Calculamos também a razão das chances para estes indicadores (*odds ratio*), onde verificamos as chances de projetos receptivos e não receptivos possuírem um determinado indicador categórico ($OR > 1$). Quanto maior a razão das chances obtidas no teste de Fisher, maiores as chances daquele grupo de projetos possuírem o respectivo indicador. Além das relações para os projetos coletados, uma análise dos resultados foi realizada entre os projetos, de modo a possibilitar a discussão do que foi encontrado sobre diferentes perspectivas, como mostra a seção a seguir.

Análise dos resultados

Além de correlacionar os indicadores com a variável dependente e investigar as relações entre os indicadores, os projetos foram agrupados também de acordo com quatro categorias: domínio, linguagem, tipo de proprietário e anos de atividade. O objetivo desta análise foi que, ao dividir os projetos em grupos, mais e menos receptivos, fosse possível visualizar com maior clareza a presença de indicadores de receptividade para cada uma das distribuições, além de avaliar, por exemplo, como as distribuições se comportam de acordo com linguagem, tipo de proprietário, domínio e idade.

Agrupamento do projeto por linguagem: A *GitHub API* fornece para cada projeto de software livre, a linguagem predominante nos arquivos do repositório de código do projeto. Buscando agrupar os projetos por linguagem, coletamos esta informação, e organizamos os projetos receptivos e não receptivos entre as respectivas linguagens de programação as quais eles predominantemente pertencem. Para cada linguagem, distribuições contendo um conjunto de projetos foi estabelecida. Como as linguagens foram definidas na seleção dos projetos, trinta projetos para cada uma das linguagens de programação foram categorizados.

Agrupamento do projeto por anos de atividade: A idade de cada projeto também foi utilizada afim de gerar compreensões sobre a receptividade dos projetos coletados. Para que fosse possível distribuir os projetos de acordo com seus respectivos anos de atividade, a idade de cada um dos projetos foi coletada por intermédio da *GitHub API*. Projetos com idades próximas fizeram parte de um único intervalo durante esta classificação.

Agrupamento do projeto por domínio: O domínio refere-se a qual área do desenvolvimento de software os projetos selecionados se encontram, áreas como *Software de aplicação*, *Framework* e *Documentação* foram utilizadas. Infelizmente, não foi possível coletar esta informação via *GitHub API* visto que a própria não disponibilizou este dado durante o período de execução do método. Por este motivo, a organização por domínio foi realizada manualmente por intermédio do próprio pesquisador.

Agrupamento do projeto por tipo de proprietário: Um projeto propenso a receptividade de novos contribuidores é aquele que, em teoria, retém um maior número de novatos. No *GitHub*, os projetos de software livre podem ser criados tanto por usuários como por organizações[5]. Estas organizações usualmente são criadas por comunidades estabelecidas por um grupo de desenvolvedores. Neste agrupamento, buscamos entender se projetos criados por usuários tendem a receber mais contribuidores do que projetos criados por organizações, e vice-versa. O tipo de proprietário de cada repositório foi coletado através da própria *GitHub API*, variando apenas entre organização ou usuário.

[5] <https://help.github.com/articles/differences-between-user-and-organization-accounts/>

Capítulo 4

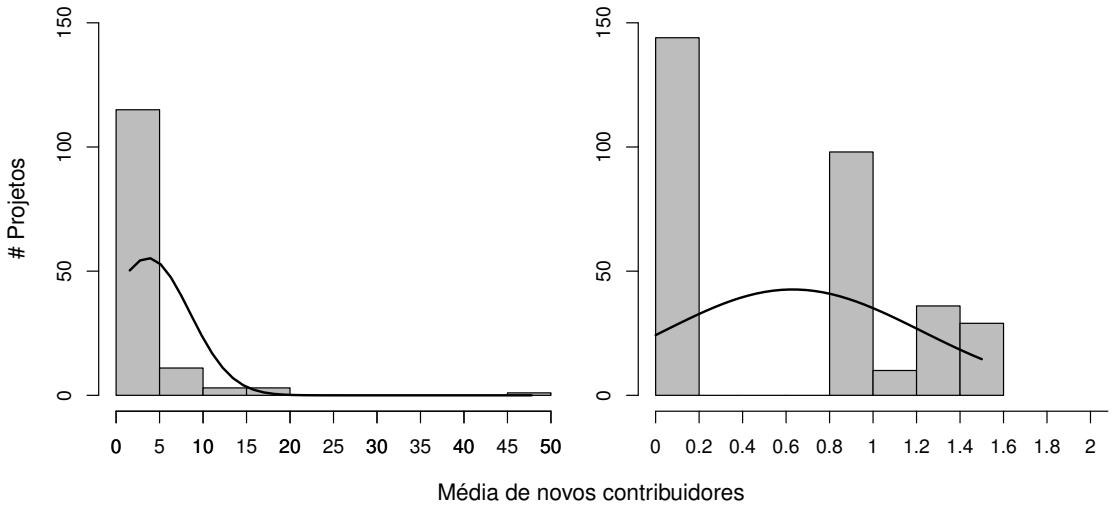
Resultados

Neste capítulo, discutimos os resultados obtidos durante a execução desta pesquisa, bem como apresentamos as análises e conclusões adquiridas de acordo com o que foi gerado neste processo. Começaremos apresentando uma análise geral da variável dependente, seguida da correlação com indicadores de receptividade definidos no capítulo anterior. Em sequência, apresentaremos a divisão dos projetos de acordo com um conjunto de categorias as quais eles pertencem. Por fim apresentaremos uma relação entre os indicadores de receptividade para projetos com maior e menor receptividade.

4.1. Análise da receptividade de projetos por meio da média de novos contribuidores semanais

Nossa variável dependente, definida como a média de novos contribuidores semanais por projeto, para um intervalo das últimas 14 semanas, foi responsável por definir a receptividade dos projetos selecionados. Projetos acima da média compuseram o grupo dos mais receptivos, enquanto que projetos abaixo da média, os menos receptivos. Entre os 450 projetos coletados, 133 ficaram acima da média de receptividade (>= 1.528 novatos por semana), à medida que os 317 demais se mantiveram abaixo da média (< 1.528 novatos por semana).

A Figura 4.1 apresenta histogramas da distribuição das médias de novos contribuidores por projeto. No grupo dos projetos com maior receptividade, apesar do conjunto de projetos estarem acima da média, o número de novatos que ingressaram nestes projetos por semana variou em grande parte entre 1 e 10 novos contribuidores, com uma diminuição na frequência de projetos conforme o número de novatos aumenta. Já os projetos menos receptivos, apesar de não ultrapassarem a marca de 2 novos contribuidores por semana, compreenderam uma variedade maior de médias de novos contribuidores por projeto, como mostra a curva desta distribuição.



(a) Projetos mais receptivos (b) Projetos menos receptivos

Figura 4.1. Histogramas relacionados ao número de novos contribuidores que projetos receptivos e menos receptivos recebem. A linha em preto representa a curva normal da respectiva distribuição.

Conforme mostra a Tabela 4.1, a média de novos contribuidores dos projetos mais e menos receptivos, quando correlacionada com indicadores de receptividade numéricos, apresenta significativa correlação pelo método de Spearman, com $p < 0{,}01$ para maioria dos resultados. Isto ocorre tanto para as médias de requisições abertas e fechadas, quanto para as contribuições e cópias dos repositórios. As contribuições fechadas e as estrelas do repositório possuem crescimento semelhante em ambas as distribuições, com maior e menor receptividade, de modo a não apresentar significativa diferença de crescimento entre as correlações.

Tabela 4.1. Resultados da correlação pelo método de Spearman (em ρ), entre os indicadores de receptividade e a média de novos contribuidores semanais para as distribuições.

Indicadores	**Projetos**	
	Mais receptivos	**Menos receptivos**
Média do número de requisições abertas por mês	0,484	0,153
Média do número de requisições fechadas por mês	0,498	0,268
Média do número de requisições aceitas por mês	0,313	0,378
Média do número de contribuições por mês	0,537	0,367
Média do número de estrelas por mês	0,311	0,292
Média do número de cópias por mês	0,495	0,277

Frente aos dados apresentados, o número de projetos para cada uma das distribuições

são significativos em termos de características e médias de novos contribuidores. Por este motivo, acreditamos também que, nesta pesquisa caberia um estudo destes projetos, afim de compreender quais características definem suas respectivas receptividades. Na seção a seguir, organizamos os projetos com maior e menor receptividade de acordo com um conjunto de valores, tendo como objetivo explorar melhor as categorias aos quais os projetos pertencem.

4.2. Análise da receptividade dos projetos por intermédio de categorias

Nesta segunda seção, os projetos selecionados foram agrupados em quatro categorias: linguagem de programação, anos de atividade, domínio e tipo de proprietário. A linguagem de programação diz respeito aquela predominante nos arquivos do repositório do projeto, a idade tem relação com os anos em que o projeto esteve em atividade, o domínio refere-se a área de desenvolvimento a qual o projeto pertence, e o tipo de proprietário define se o proprietário é um desenvolvedor (usuário) ou uma organização (comunidade). Apresentamos abaixo as observações analisadas para cada uma destas quatro categorias, onde avaliamos em quais grupos os projetos mais e menos receptivos se enquadram.

4.2.1. Agrupamento dos projetos por linguagem

Para as linguagens de programação, já havíamos avaliado na seção 3.2.2 como os projetos se organizavam em números de contribuidores, mas não avaliamos como os projetos se distribuem quanto a variável dependente, definida como média de novos contribuidores semanais. Neste caso, as distribuições com maior e menor receptividade apresentaram diferenças entre as linguagens de programação. Projetos escritos em `Go`, `Python`, `Ruby` e `Javascript` apareceram com maior frequência no grupo dos mais receptivos, ao mesmo passo que a linguagem com maior frequência no grupo dos menos receptivos, `Perl`, sequer apareceu no grupo dos projetos com maior receptividade. `Perl` foi seguida em ordem de frequência por `Objetive-C` e `CoffeScript`, que aparecem com as menores frequências do grupo mais receptivo. Estas comparações podem ser vistas na Figura 4.2.

As razões pelas quais os projetos se organizam de acordo com as respectivas linguagens ainda é desconhecido. Todavia, acreditamos que um dos potenciais fatores que influencia a receptividade destas linguagens é a popularidade. As quatro primeiras linguagens que apareceram com maior frequência nos projetos receptivos se enquadram também entre as cinco linguagens mais populares em termos de requisições externas, segundo o website GitHut[1].

[1] <https://madnight.github.io/githut/>

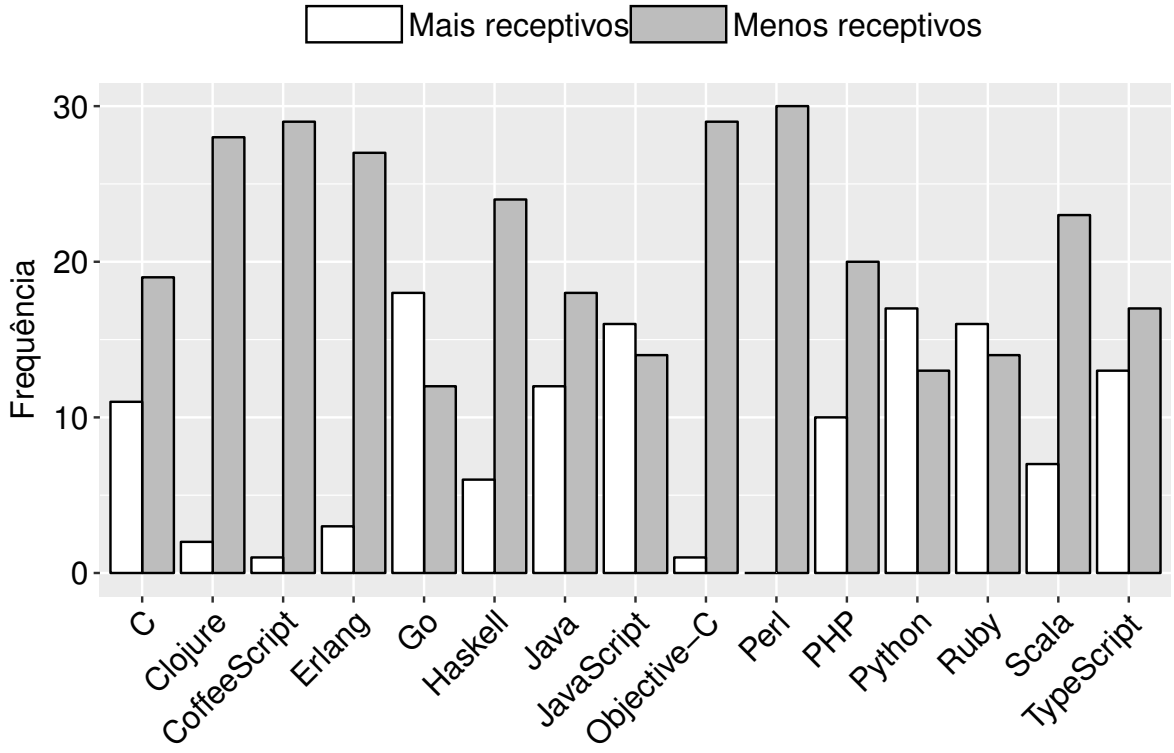


Figura 4.2. Frequência de projetos de acordo com as linguagens de programação. Projetos mais receptivos em branco, menos receptivos em cinza.

4.2.2. Agrupamento dos projetos por tipo de proprietário

Na plataforma *GitHub*, os projetos podem ser de propriedade de um desenvolvedor (usuário) ou de uma organização (comunidade). Entre os projetos receptivos avaliados nesta pesquisa, 105 dos 133 projetos são mantidos e administrados por comunidades, resultando em 78 pontos percentuais do total deste grupo. Nos projetos com menor receptividade, essa diferença é um pouco menor, onde projetos mantidos por organizações compõem 60 pontos percentuais deste grupo. Levando em consideração que os projetos receptivos nesta pesquisa são geridos 18 pontos percentuais a mais por organizações, acreditamos que o tipo de proprietário possui sim alguma relação com a receptividade, entretanto, este fenômeno precisa ser melhor estudado, afim de compreender as razões que levam projetos de comunidades a reterem mais contribuidores.

4.2.3. Agrupamento dos projetos por anos de atividade

Os projetos utilizados nesta pesquisa também foram agrupados por anos de atividade. De modo geral, os projetos selecionados compreenderam idades entre menos de 1 ano até 9 anos de idade. Entretanto, apesar da quantidade de projetos distribuídos entre as idades ser significativa, os números apresentados são razoavelmente similares para projetos menos e mais receptivos. A média de idade dos projetos receptivos é de 4 anos, ao passo que dos

menos receptivos é de 5 anos. Nas Figuras 4.3 e 4.4, é possível observar a distribuição para ambos os grupos de projetos, ao qual as observações dispostas acima podem ser confirmadas. Consideramos desta maneira, que para os projetos coletados, os anos de atividade não possuem relação com a receptividade dos mesmos.

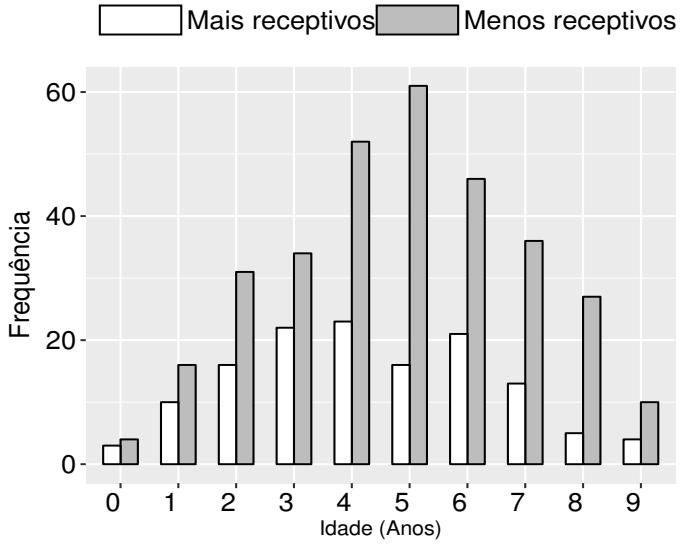


Figura 4.3. Histogramas sobrepostos, relativos a idade de cada projeto. Em cinza, projetos com maior receptividade, em branco, com menor.

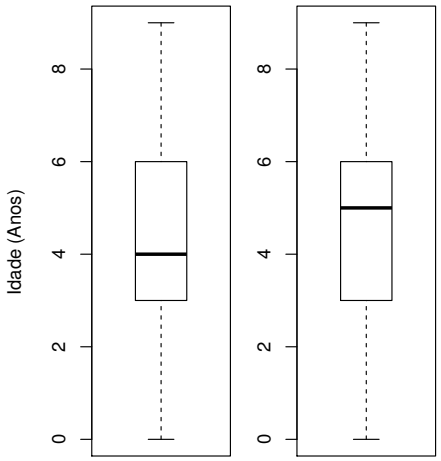


Figura 4.4. Distribuição de idade para os projetos mais e menos receptivos. A esquerda, projetos mais receptivos, a direita, menos.

4.2.4. Agrupamento dos projetos por domínio

Os projetos ao serem agrupados por domínio, foram definidos entre seis categorias baseadas no trabalho de Borges et al. (2016b): `Aplicação`, `Documentação`, `Ferramenta`, `Framework ou Biblioteca (Não-web)`, `Framework ou Biblioteca (Web)`, `Programas do sistema`. Aqueles em que não foi possível definir uma das seis categorias, receberam o rótulo de `Sem categoria`. A Figura 4.5 apresenta a distribuição de domínios para projetos com maior e menor receptividade.

É possível observar que a distribuição dos projetos é similar para ambos os grupos, *frameworks* e bibliotecas lideram a área dos projetos, bem como as ferramentas. Neste conjunto de projetos, não foi possível observar nenhuma grande diferença entre receptividade e a área as quais os projetos pertencem. Um dos grandes agravantes que interferiram nesta agrupamento, está no fato de que nem todos os projetos coletados puderam ser classificados entre os domínios estabelecidos.

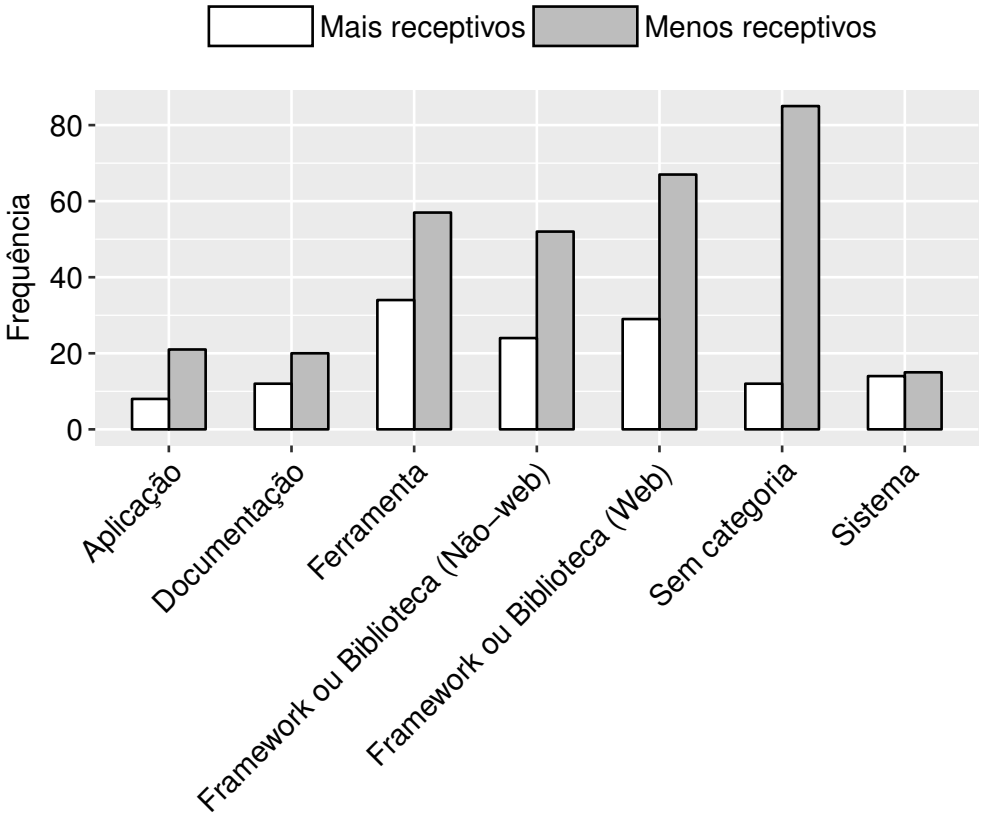


Figura 4.5. Frequência de projetos de acordo com domínio de atuação. Projetos mais receptivos em branco, menos receptivos em cinza.

4.3. Relação dos indicadores de receptividade com base na média de novos contribuidores por semana

A terceira e última etapa de nossa pesquisa concentrou-se em relacionar as variáveis independentes apresentadas na Tabela 3.1, conhecidas como indicadores de receptividade, utilizando como base a variável dependente definida, afim de encontrar relações entre a entrada de novos contribuidores e algumas das características que cada um dos projetos possuí.

Para estas relações, mantivemos os projetos divididos de acordo com a média de receptividade e então relacionamos o conjunto de indicadores entre os grupos mais e menos receptivos, confrontando, por exemplo, o indicador de contribuições dos projetos mais receptivos, com o indicador de contribuições dos projetos com menor receptividade. Apresentamos abaixo um conjunto de observações empreendidas com a comparação dos indicadores, afim de entender a dimensão das diferenças.

As estatísticas de um projeto podem estar relacionadas a sua receptividade. Quando avaliamos as relações entre os indicadores numéricos, para os projetos mais e

menos receptivos, observamos que estes apresentam uma relação alta para a grande maioria dos indicadores. Em outras palavras, o número de novos contribuidores que um determinado projeto recebe, pode estar relacionado a valores que este projeto apresenta, como a média de contribuidores, estrelas e cópias recebidas pelo projeto ao longo dos meses. Quanto as requisições abertas (*opened*), o tamanho do efeito foi considerado alto, e as distribuições aceitas como independentes ($p < 0{,}05$, $delta = 0{,}527$). O mesmo aconteceu para as requisições fechadas (*closed*) ($p < 0{,}05$, $delta = 0{,}649$) e aceitas (*merged*) ($p < 0{,}05$, $delta = 0{,}705$).

Para as contribuições (*commits*), os números também apresentaram um efeito alto entre as distribuições ($p < 0{,}05$, $delta = 0{,}689$), o mesmo ocorreu para estrelas ($p < 0{,}05$, $delta = 0{,}489$) e cópias ($p < 0{,}05$, $delta = 0{,}544$). Deste modo, os resultados obtidos na comparação entre os indicadores numéricos, para ambas as distribuições sugerem que as requisições externas, contribuições e a média de cópias e estrelas por mês, podem definir a receptividade de um projeto.

Tabela 4.2. Valores de p e *delta* obtidos na relação da variável dependente com os indicadores numéricos, por meio dos métodos Mann-Whitney-Wilcoxon e delta de Cliff

Indicadores	**Estatísticas do projeto**	
	p	**delta**
Média do número de requisições abertas por mês	<0.05	0.527
Média do número de requisições fechadas por mês	<0.05	0.649
Média do número de requisições aceitas por mês	<0.05	0.705
Média do número de contribuições por mês	<0.05	0.689
Média do número de estrelas por mês	<0.05	0.489
Média do número de cópias por mês	<0.05	0.544

A retenção de novos contribuidores pode depender do modo como estes são orientados. Uma diversidade de recursos usualmente disponíveis em projetos de software livre buscam auxiliar, por meio de informações, novos contribuidores a realizarem suas primeiras contribuições. Entre estes recursos estão três indicadores estudados nesta pesquisa: *README.md*, *CONTRIBUTING.md* e *Wiki*. Entretanto, com base no teste exato de Fisher aplicado, apenas as chances do arquivo *CONTRIBUTING.md* existir em projetos receptivos foram apresentadas como as mais relevantes ($p < 0{,}01$, OR = 2,767), ao passo que o arquivo *README.md* ($p = 0{,}527$, OR = 0,818) e o recurso *Wiki* ($p < 0{,}01$, OR = 0,518) não satisfizeram todas as condições necessárias. É possível dizer que apenas o arquivo *CONTRIBUTING.md* tem maiores chances de aparecer em projetos com maior receptividade,

e estes resultados, em especial a razão das chances (*odds ratio*), sugerem que a utilização do arquivo *CONTRIBUTING.md* é 2,767 vezes mais efetiva na retenção de novos contribuidores, em comparação com os outros indicadores analisados nesta categoria. A Tabela 4.3 apresenta um exemplo de tabela de contingência utilizada na aplicação do método de Fisher nesta pesquisa.

Tabela 4.3. Tabela de contingência (2 × 2) para o indicador *README.md*

Indicador numérico	**Projetos Mais receptivos**	**Projetos Menos receptivos**
Contém *README.md*	115	281
Não contém *README.md*	18	36

As tecnologias utilizadas em um projeto parecem não possuir relação com receptividade. Ao contribuir com projetos de software livre, é comum encontrarmos uma diversidade de tecnologias utilizadas que facilitam o desenvolvimento dos códigos em questão. Entre estas tecnologias, estão o sistema de Gerenciador de tarefas (*Issue Tracker*) e um sistema desenvolvido pelo *GitHub*, conhecido como *Project Board*. Apesar destas tecnologias usualmente contribuírem para o desenvolvimento dos projetos, estas não apresentaram chances significativas de aparecerem em projetos mais receptivos, o que pode pressupor que a utilização destas duas tecnologias não necessariamente contribuem para a retenção de novos contribuidores. Estas observações podem ser visualizadas a partir dos resultados obtidos no teste exato de Fisher, para o Gerenciador de tarefas ($p < 0{,}01$, OR = 0,518) e o *Project Board* ($p < 0{,}01$, OR = 0,361), que não corresponderam ao critério necessário (OR > 1).

4.4. Ameaças à validade

Como ameaças à validade, destacamos que, embora os indicadores de receptividade apresentados nesta pesquisa, quando relacionados por intermédio da variável dependente, apresentem altos índices de tamanho de efeito, a causalidade entre estes indicadores ainda precisa ser melhor estudada, afim de compreender com maior clareza se de fato estas relações estão ligadas aos contribuidores que os projetos retém. Acreditamos que o critério para divisão entre projetos mais e menos receptivos, definido como a média de novos contribuidores semanais para as últimas 14 semanas coletadas, precisa também ser melhor investigado, de modo a compreender se estes valores de fato podem ser aplicados como critério divisor de projetos mais e menos receptivos. Salientamos por último que os indicadores utilizados nesta pesquisa são apenas algumas das características que os projetos possuem, ao passo que outros valores dentro de um projeto podem também interferir em sua receptividade, como as licenças de software utilizadas, o uso de mecanismos de integração contínua e o modo como os contribuidores recebem novos contribuidores.

Capítulo 5

Conclusão

A retenção de novos contribuidores em projetos de software livre é fator crucial para contínua existência dos mesmos no âmbito do desenvolvimento de software. Permitir e contribuir para que novos contribuidores ingressem nestas comunidades é tarefa indispensável para os desenvolvedores dos projetos. É importante também que novos contribuidores tenham a oportunidade de ingressar em projetos de software livre que de fato almejem receber novos desenvolvedores. Com o propósito de contribuir para receptividade de novatos, buscamos nesta pesquisa avaliar, a partir de um conjunto de indicadores, relações entre a recente receptividade dos projetos coletados, e os valores que eles apresentam no decorrer do tempo em que estão ativos.

Entre as descobertas, foi possível observar, por exemplo, que indicadores numéricos, como a média de cópias por mês, estão fortemente relacionados a receptividade de novatos nas últimas semanas. Observamos que nem todos os arquivos e recursos utilizados nos projetos, como o sistema de *Gerenciador de tarefas*, possuem alguma relação com a receptividade dos mesmos, e a medida que as linguagens de programação utilizadas nos repositórios coletados definiram suas recentes receptividades, a idade, a área de atuação e o proprietário, nem tanto. Consideramos que é possível definir a receptividade de projetos de software livre com o uso de um conjunto de indicadores. Todavia, para que isto seja possível, é necessário que os indicadores sejam bem estabelecidos, visto que uma diversidade de fatores podem influenciar a veracidade dos mesmos e, por consequência, suas relações com a retenção de novos contribuidores.

Para trabalhos futuros, desejamos avaliar a relação que outras características presentes nos projetos de software livre possuem com a quantidade de novos contribuidores que os projetos recebem, afim de definir novos indicadores de receptividade. Desejamos também estabelecer novas estratégias para contribuir com a entrada de novatos em projetos de software livre, como a criação de um mecanismo de recomendação de projetos receptivos aos novatos.

Referências

BEN, X.; BEIJUN, S.; WEICHENG, Y. Mining developer contribution in open source software using visualization techniques. In: *2013 Third International Conference on Intelligent System Design and Engineering Applications*. [S.l.: s.n.], 2013. p. 934–937.

BORGES, H.; HORA, A.; VALENTE, M. T. Understanding the factors that impact the popularity of github repositories. In: *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. [S.l.: s.n.], 2016. p. 334–344.

BORGES, Hudson; HORA, Andre; VALENTE, Marco Tulio. Understanding the factors that impact the popularity of github repositories. *arXiv preprint arXiv:1606.04984*, 2016.

CANFORA, Gerardo; PENTA, Massimiliano Di; OLIVETO, Rocco; PANICHELLA, Sebastiano. Who is going to mentor newcomers in open source projects? In: ACM. *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*. [S.l.], 2012. p. 44.

CAPILUPPI, Andrea; MICHLMAYR, Martin. From the cathedral to the bazaar: An empirical study of the lifecycle of volunteer community projects. In: _____. *Open Source Development, Adoption and Innovation: IFIP Working Group 2.13 on Open Source Software, June 11–14, 2007, Limerick, Ireland*. Boston, MA: Springer US, 2007. p. 31–44. ISBN 978-0-387-72486-7. Disponível em: <http://dx.doi.org/10.1007/978-0-387-72486-7_3>.

COELHO, Jailton; VALENTE, Marco Tulio. Why modern open source projects fail. In: *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. New York, NY, USA: ACM, 2017. (ESEC/FSE 2017), p. 186–196. ISBN 978-1-4503-5105-8.

CROWSTON, Kevin; HOWISON, James. The social structure of free and open source software development. *First Monday*, v. 10, n. 2, 2005.

DABBISH, Laura; STUART, Colleen; TSAY, Jason; HERBSLEB, Jim. Social coding in github: transparency and collaboration in an open software repository. In: ACM. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. [S.l.], 2012. p. 1277–1286.

FORWARD, Andrew; LETHBRIDGE, Timothy C. The relevance of software documentation, tools and technologies: A survey. In: *Proceedings of the 2002 ACM Symposium on Document Engineering*. New York, NY, USA: ACM, 2002. (DocEng '02), p. 26–33. ISBN 1-58113-594-7.

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.

GOUSIOS, Georgios; STOREY, Margaret-Anne; BACCHELLI, Alberto. Work practices and challenges in pull-based development: The contributor's perspective. In: *Proceedings of the 38th International Conference on Software Engineering*. New York, NY, USA: ACM, 2016. (ICSE '16), p. 285–296. ISBN 978-1-4503-3900-1.

HANNEBAUER, Christoph; BOOK, Matthias; GRUHN, Volker. An exploratory study of contribution barriers experienced by newcomers to open source software projects. In: ACM. *Proceedings of the 1st International Workshop on CrowdSourcing in Software Engineering*. [S.l.], 2014. p. 11–14.

HARS, Alexander; OU, Shaosong. Working for free? motivations of participating in open source projects. In: IEEE. *System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on*. [S.l.], 2001. p. 9–pp.

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.

HIPPEL, Eric Von. Innovation by user communities: Learning from open-source software. *MIT Sloan management review*, Massachusetts Institute of Technology, Cambridge, MA, v. 42, n. 4, p. 82, 2001.

JENSEN, Carlos; KING, Scott; KUECHLER, Victor. Joining free/open source software communities: An analysis of newbies' first interactions on project mailing lists. In: IEEE. *System Sciences (HICSS), 2011 44th Hawaii International Conference on*. [S.l.], 2011. p. 1–10.

KELTY, Christopher M. Free software/free science. *First Monday*, Valauskas, Edward J., v. 6, n. 12, 2001.

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.

LETHBRIDGE, Timothy C; SINGER, Janice; FORWARD, Andrew. How software engineers use documentation: The state of the practice. *IEEE software*, IEEE, v. 20, n. 6, p. 35–39, 2003.

MADEY, Gregory; FREEH, Vincent; TYNAN, Renee. The open source software development phenomenon: An analysis based on social network theory. *AMCIS 2002 Proceedings*, p. 247, 2002.

MOURA, I.; PINTO, G.; EBERT, F.; CASTOR, F. Mining energy-aware commits. In: *MSR*. [S.l.: s.n.], 2015. p. 56–67.

PINTO, Gustavo; STEINMACHER, Igor; GEROSA, Marco. More common than you think: An in-depth study of casual contributors. In: *SANER*. [S.l.: s.n.], 2016. p. 112–123.

QURESHI, Israr; FANG, Yulin. Socialization in open source software projects: A growth mixture modeling approach. *Organizational Research Methods*, SAGE Publications Sage CA: Los Angeles, CA, v. 14, n. 1, p. 208–238, 2011.

RAY, Baishakhi; POSNETT, Daryl; FILKOV, Vladimir; DEVANBU, Premkumar. A large scale study of programming languages and code quality in github. In: ACM. *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. [S.l.], 2014. p. 155–165.

ROMANO, Jeanine; KROMREY, Jeffrey D; CORAGGIO, Jesse; SKOWRONEK, Jeff. Should we really be using t-test and cohen's d 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.

SIEGEL, Sidney; JR, N John Castellan. *Estatística não-paramétrica para ciências do comportamento*. [S.l.]: Artmed Editora, 1975.

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

STEINMACHER, Igor; CONTE, Tayana; GEROSA, Marco Aurélio; REDMILES, David. Social barriers faced by newcomers placing their first contribution in open source software projects. In: ACM. *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing*. [S.l.], 2015. p. 1379–1392.

STEINMACHER, Igor; GEROSA, Marco Aurélio; REDMILES, D. Attracting, onboarding, and retaining newcomer developers in open source software projects. In: *Workshop on Global Software Development in a CSCW Perspective*. [S.l.: s.n.], 2014.

STEINMACHER, Igor; SILVA, Marco Aurélio Graciotto; GEROSA, Marco Aurélio. Barriers faced by newcomers to open source projects: a systematic review. In: SPRINGER. *IFIP International Conference on Open Source Systems*. [S.l.], 2014. p. 153–163.

STEINMACHER, Igor; SILVA, Marco Aurelio Graciotto; GEROSA, Marco Aurelio; REDMILES, David F. A systematic literature review on the barriers faced by newcomers to open source software projects. *Information and Software Technology*, Elsevier, v. 59, p. 67–85, 2015.

STEINMACHER, I.; WIESE, I.; CHAVES, A. P.; GEROSA, M. A. Why do newcomers abandon open source software projects? In: *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*. [S.l.: s.n.], 2013. p. 25–32.

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.

TIAN, Y.; NAGAPPAN, M.; LO, D.; HASSAN, A. E. What are the characteristics of high-rated apps? a case study on free android applications. In: *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. [S.l.: s.n.], 2015. p. 301–310.

TSAY, Jason; DABBISH, Laura; HERBSLEB, James. Influence of social and technical factors for evaluating contribution in github. In: *ICSE*. [S.l.: s.n.], 2014. p. 356–366.

UPTON, Graham JG. Fisher's exact test. *Journal of the Royal Statistical Society. Series A (Statistics in Society)*, JSTOR, p. 395–402, 1992.

WU, Ming-Wei; LIN, Ying-Dar. Open source software development: an overview. *Computer*, IEEE, v. 34, n. 6, p. 33–38, 2001.

YE, Yunwen; KISHIDA, Kouichi. Toward an understanding of the motivation open source software developers. In: IEEE COMPUTER SOCIETY. *Proceedings of the 25th international conference on software engineering*. [S.l.], 2003. p. 419–429.

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.