HelpBot, Atribuidor e Agregador: Projetando e Implementando um Robô Social para Programadores na Plataforma GitHub

Douglas Castro[1], **Lesandro Ponciano**[1]

[1]Pontifícia Universidade Católica de Minas Gerais (PUC MINAS)
Belo Horizonte - MG - Brasil

douglaswcastro@hotmail.com, lesandrop@pucminas.br

Resumo. *Há um crescente uso de robôs para automatização de tarefas antes feitas manualmente. As tarefas executadas por esses robôs geram um ganho no tempo de execução. A engenharia de software é marcada pelo uso de ferramentas de automatização, mas nem tanto por robôs. Nos deparamos com a falta de informação sobre como os robôs podem ser aplicados para automatizar tarefas e auxiliar programadores em um ambiente de desenvolvimento baseado em repositórios de código. Nesse contexto, projetou-se e implementou um robô para o auxílio de desenvolvedores na plataforma de código GitHub. O robô foi desenvolvido utilizando a linguagem de programação Python, utilizando API do Github para coletar os dados necessários, possuindo duas funcionalidades, de atribuição e agregação. O robô foi desenvolvido e configurado para responder as requisições dos usuários no GitHub. Após o desenvolvimento, ele foi avaliado por cinco desenvolvedores que o testaram realizando as requisições sobre tópicos que tinham interesse. Com os testes foi aplicado um questionário com três perguntas para os desenvolvedores que avaliaram o robô positivamente. Indicando que o usariam e recomendariam para outros desenvolvedores caso precisem das funcionalidades que ele traz.*

1. Introdução

Os robôs, no contexto da Engenharia de *Software*, podem ser definidos como programas desenvolvidos para auxiliarem em tarefas repetitivas de uma forma similar a um humano [Dunham and Melnick 2008]. Robôs automatizam tarefas de modo que elas são executadas de forma mais eficiente, em maior escala, em menor tempo e sem a intervenção humana [Kollanyi 2016]. Com o auxílio dessa tecnologia, é possível realizar as tarefas em um tempo menor do que se fossem executadas manualmente. Com isso, as tarefas tornaram-se cada vez mais automatizadas, gerando um ganho no tempo de execução. A demanda para o desenvolvimento de robôs com a finalidade de facilitar a execução de tarefas está cada vez mais requisitada e em várias áreas está ocorrendo essa crescente demanda. Um exemplo é na área jurídica, que está implementando robôs para protocolar processos, automatizando e agilizando tarefas que antes eram feitas manualmente e demandava muito tempo [Nunes 2018].

Outra área na qual robôs podem ser aplicados é a Engenharia de *Software*. Há diversas tarefas no ambiente de desenvolvimento de *software* que podem ser automatizadas com o uso de robôs. No entanto, há falta de informação sobre robôs nesse contexto, tornando mais difícil avaliar a sua eficiência e como agregar valor com a sua utilização.

Por exemplo, existe também essa mesma falta de informações sobre como robôs podem auxiliar ao se atribuir tarefas para programadores nos repositórios de código, como aqueles mantidos em plataformas como o *GitHub*. Além disso, como robôs podem agregar informações dos repositórios e fornecê-las como utilidades aos programadores. Isso gera o problema da falta de informação sobre como robôs podem ser aplicados para automatizar tarefas e auxiliar os programadores em um ambiente de desenvolvimento baseado em repositórios de código. Esse é, portanto, o problema a ser tratado neste trabalho. A importância de existir informações sobre a eficiência e como implementar robôs na plataforma *GitHub* é de suma importância para automatização de tarefas na plataforma.

O objetivo geral deste trabalho é implementar um robô para o *GitHub* de modo a auxiliar programadores em tarefas de agregação e atribuição. Para alcançar o objetivo principal almeja-se atingir os seguintes objetivos específicos: i) propor e implementar a atribuição; ii) propor e implementar a agregação; iii) compreender a eficácia de robôs no contexto do *GitHub*. A agregação é a funcionalidade que verifica os repositórios públicos de outros desenvolvedores que tenham alguma relação com quem solicita a funcionalidade do robô, gerando uma lista com os desenvolvedores que têm algum repositório público relacionado, por exemplo, com a linguagem que foi passada para o robô realizar a busca. Atribuição é a funcionalidade que busca o desenvolvedor na rede de amizade de quem solicitou que tem alguma correspondência com o tópico pesquisado, por exemplo,desenvolvedor que tem maior quantidade de repositórios em alguma determinada linguagem.

A Seção 2 apresenta o referencial teórico usado neste trabalho. A Seção 3 descreve os trabalhos relacionados relevantes a este trabalho. A Seção 4 apresenta a proposta e informações sobre a implementação do robô *HelpBot*. A seção descreve os testes realizados no *HelpBot* e a avaliação dos desenvolvedores, juntamente com as discussões e implicações do trabalho. Na Seção 6 é apresentada a conclusão deste trabalho.

2. Referencial Teórico

O referencial teórico está dividido em três tópicos, em que se aborda os principais conceitos tratados neste trabalho e que são essenciais para o melhor entendimento. São eles: repositórios de código; suporte ferramental para engenharia de software e robôs.

2.1. Repositório de código

Repositórios de códigos são usados para gerenciar o progresso do desenvolvimento e controle de versões de projetos, tendo um melhor controle e previsão, gerando um histórico com todo o código fonte e alterações [D'Ambros et al. 2008]. Um exemplo de plataforma que permite manter repositórios é o *GitHub*. O *GitHub* é um sistema colaborativo para hospedagem de código, construído sobre o controle de versão do *Git* [Kalliamvakou et al. 2014]. O controle de versão *Git* registra as mudanças realizadas no código fonte de um projeto, de forma que é possível recuperar alguma versão específica do projeto caso seja necessário, também podendo revisar alterações feitas no código fonte.

O *GitHub* é o maior sistema de hospedagem de códigos do mundo, com mais de 10,6 milhões de repositórios em janeiro de 2014 [Kalliamvakou et al. 2014]. Ele disponibiliza os seus dados por meio de uma Interface de Programação de Aplicações (API, do inglês *Application Programming Interface*) acessível para desenvolvedores, deixando-o

aberto para criação de robôs para automatizar a coleta e publicação de dados. Os repositórios do *GitHub* também podem ser usados para outras finalidades que não o desenvolvimento de software colaborativo [Kalliamvakou et al. 2014], como para hospedar projetos pessoais, sem planos de colaborar em seu trabalho. As vantagens do *GitHub* são as funções sociais, em que os usuários podem seguir trabalhos de outros, reconhecer programas escritos por outros desenvolvedores, criar cópias do repositório de outro usuário e trabalhar nesta versão [Kollanyi 2016].

2.2. Suporte Ferramental para Engenharia de Software

Ferramentas do tipo Engenharia de Software Assistida por Computador (CASE, do inglês *Computer Aided Software Engineering*) visam aumentar a produtividade e a qualidade dos artefatos de software [Fidalgo et al. 2015]. Ferramentas CASE são programas de apoio às atividades do processo de desenvolvimento de software e podem ser classificadas em funcionais ou baseadas em atividades [Simão and Nomura 2017]. Nas duas classificações das ferramentas CASE se define o conceito de ferramenta de documentação: ferramentas para o auxílio à criação de documentos relacionados ao processo de desenvolvimento de software, como especificação, manuais, relatórios [Simão and Nomura 2017].

Os Ambientes Integrados de Desenvolvimento (IDEs, do inglês *Integrated Development Environment*) possuem uma interface amigável para o desenvolvimento de programas. Elas reúnem características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar o processo [Lopes and Garcia 2002]. Os componentes básicos de uma IDE são: editor, compilador, *linker* e *debugger* [Lopes and Garcia 2002]. Uma das principais vantagens das IDEs são os *debuggers*, que apontam os erros que podem ocorrer ao escrever o código.

No processo de desenvolvimento de software, mudanças ocorrem a todo momento. Caso essas mudanças não sejam devidamente documentadas e comunicadas, poderão acarretar problemas [Nunes 2005]. A necessidade de versionamento surgiu principalmente em grandes projetos, em que era preciso registrar a inclusão, modificação ou exclusão de códigos ou funcionalidades [Nogueira et al. 2017]. Existem diversos sistemas de controle de versão como o Sistema de Versões Concorrentes (*CVS*, do inglês *Concurrent Version System*), Apache *Subversion* (*SVN*) e o *Git*.

2.3. Robôs

No contexto de Engenharia de Software, um robô é uma aplicação de software que executa tarefas, a uma taxa mais elevada do que seria possível para um humano [Freitas 2014]. Um conjunto de robôs conectados à Internet que se comunicam com a finalidade de executar uma tarefa em comum é denominado de *Botnet*. Uma quantidade crescente de código-fonte aberto está disponível na Internet para configuração rápida de robô no Twitter [Kollanyi 2016].

Uma quantidade crescente de código-fonte aberto está disponível na Internet para configuração rápida e implantar robôs no *Twitter* [Kollanyi 2016]. Esses robôs geralmente usam APIs para se comunicarem e executarem as ações para as quais eles foram criados, realizando chamadas para essas APIs. Chamadas de APIs precisam seguir uma sintaxe específica, que possui uma extensa documentação sobre como conectar na do *GitHub*. Nos últimos anos, o número de robôs no *Twitter* quadruplicou e em abril de 2016 existiam mais de 4.000 repositórios no *GitHub* [Kollanyi 2016].

3. Trabalhos Relacionados

Nesta seção são apresentados os trabalhos relacionados a robôs e *GitHub*, mesmo eles não se tratando da aplicação de robôs diretamente no *GitHub*, eles trazem informações relevantes sobre o desenvolvimento e hospedagem de robôs.

Kollanyi (2016) examina a prática de escrever e compartilhar códigos de robôs no *GitHub*. Através da API do *GitHub* onde permite-se a obtenção de uma lista de repositórios, onde contêm pesquisas científicas em seu nome, descrição ou no leia-me. Uma chamada na API do *GitHub* forneceu 2783 resultados, outra chamada adicionando o termo *TwitterBot* retornou mais 764. Em média os usuários tinham 29 repositórios na época da coleta e eles já estavam registrados no *GitHub* dois anos antes de publicar seu primeiro código de robô. O trabalho está relacionado pois trata-se do desenvolvimento de robôs, abordando também o uso da API do *GitHub* para obtenção dados.

Braz and Goldschmidt (2018) abordam o uso de contas controladas por robôs no *Twitter* e *Facebook*, como essas contas são utilizadas para propagar notícias falsas e podendo manipular a opinião das pessoas. A detecção dos conhecidos Robôs Sociais é a identificação de contas automatizadas com o intuito de reduzir os efeitos que essas contas podem causar e diminuir o número das mesmas. Para essa análise foi proposto um método que aplica rede neural para identificar mensagens suspeitas, baseando-se nos atributos comportamentais e no percentual de mensagens depreendidas como suspeitas pelo modelo. Como resultado, mostra-se a análise do contexto textual das mensagens, diferente daqueles que se baseiam somente no comportamento das contas. Podemos relacionar o trabalho com a abordagem de robôs na automação de tarefas que são feitas manualmente.

Beschastnikh et al. (2017) tratam do estudo de robôs e os desafios que devem ser abordados para que robôs se tornem uma realidade, e quais são as formas adequadas para que eles interajam melhor com os desenvolvedores, e como eles podem ser hospedados. Estes robôs podem ser implementados por qualquer pessoa, mas são especialmente úteis para os pesquisadores que podem usa-los para análise e avaliação de suas propostas. Acreditam que os robôs, têm o potencial para acelerar a adoção de pesquisas pelos praticantes. Esse trabalho está relacionado pois ele trata diretamente do estudo e análise de desafios sobre os robôs, e como eles podem interagir com os desenvolvedores e ser hospedados.

Yu et al. (2016) faz um estudo detalhado sobre *pull request* (PR) no *GitHub*, analisando o tempo médio de uma revisão de código, e de como cada desenvolvedor tem a oportunidade de participar da revisão. Os títulos e descrições dos PRs são extraídos e indexados usando o espaço Modelo *Vector*, medindo a semelhança semântica dos PRs com os que são armazenados no histórico. Os avaliadores recomendam pois fará com que o processo de revisão seja mais eficaz. EO trabalho propõe uma nova abordagem que combine a recuperação da informação com a rede social em análise. Esse trabalho relaciona-se, pois aborda diretamente os repositórios de código no *GitHub*, mas focado nos PRs (quando envia-se uma sugestão de melhoria para o repositório), com isso ele também aborda sobre o estudo de funcionamento do *GitHub*.

Sandim et al. (2018) tratam sobre o *Truck Factor* (TF) (número de desenvolvedores que perturbariam um projeto se o abandonassem). TF é uma medida de

risco que analisa o grau de conhecimento compartilhado em um projeto de software [Hannebauer and Gruhn 2014]. Usando o Social *Truck Factor* (STF), uma abordagem social para estimar o TF a partir de uma rede de colaboração sobre o *GitHub*. O STF se baseia no ranqueamento de desenvolvedores a partir da agregação de métricas topológicas e semânticas da rede de colaboração. Esse trabalho está relacionado pois trata de um estudo direto sobre o *Github*, que é a plataforma de estudo neste trabalho, realizando também uma análise de repositórios de códigos.

Pereira and Lesandro (2019) tratam de tarefas de atribuição e agregação de valor em sistemas de computação, conceituando e exemplificando os dois tipos de tarefas que são as funcionalidades do robô proposto neste trabalho. Os autores abordam a funcionalidade de atribuição dado uma tarefa que deve ser realizada por um ser humano, dentre um conjunto de pessoas, alocar essa tarefa para uma ou mais pessoas que demonstram possuir maior capacidade, habilidade ou aptidão. Agregação dado um conjunto de respostas criadas por pessoas, analisá-las conjuntamente a fim de definir, a partir de todas as respostas, características comuns das tarefas ou dos trabalhadores.

4. Proposta e Implementação do Robô HelpBot

Atualmente quando algum desenvolvedor de *software* necessita de auxílio em algum tópico para buscar ajuda no *GitHub*, ele precisa entrar no perfil de cada desenvolvedor de *software* da sua rede. Após isso, tem que verificar todos os repositórios para descobrir se ele tem algum projeto que envolva o tópico. Isso gera tempo de pesquisa para descobrir a quem recorrer para sanar essa dúvida.

O robô proposto neste trabalho tem a finalidade de ajudar desenvolvedores de *softwares* que possuem dúvida ou dificuldade em determinado tópico. Ele ajuda a identificar outro desenvolvedor *software* da rede de contatos do desenvolvedor que fez a solicitação no *GitHub*, que possui conhecimento sobre o tópico. Realizando a busca e identificando os desenvolvedores de *softwares* que vão poder ajuda-los a sanar as respectivas duvidas. Essa proposta de robô é inspirada em trabalhos anteriores que trataram de robô para a plataforma Slack [Jardim de Oliveira and Ponciano 2018] e para a rede social Twitter [Pereira and Ponciano 2019]

4.1. Especificação e Projeto

Nesta seção serão apresentados os requisitos funcionais e não funcionais, os cenários de uso do HelpBot, trazendo as funcionalidades de atribuição e agregação e o seu funcionamento.

4.1.1. Requisitos Funcionais

Esta seção traz os requisitos funcionais do *HelpBot*. As funcionalidades do *HelpBot* foram implementadas baseadas no conceito de atribuição e agregação que são levantados por [Pereira and Ponciano 2019].

A funcionalidade de atribuição que no robô é tratada como ”Quem”é usada quando o desenvolvedor de *software* quer pesquisar sobre quem na sua rede de amizade, que ele segue no *GitHub*, possui conhecimento sobre algum tópico, que pode ser uma linguagem

de programação ou alguma tecnologia. Para isso, o HelpBot faz uma verificação com o tópico informado para ele nos repositórios públicos dos desenvolvedores que o desenvolvedor que fez a solicitação segue. Retornando o nome de usuário do desenvolvedor com mais correspondências com o tópico informado.

Algorithm 1: Pseudcódigo funcionalidade de atribuição do HelpBot

input : Requisição realizada para o robô passando os dados necessários em JSON
output: login do usuário que mais corresponde com a pesquisa informada

```
login ← douglaswcastro;      /* login do desenvolvedor que
 realização a requisição para o robô */
topico ← Python ; /* Tópico informado pelo desenvolvedor
 para realizar a pesquisa */
desenvolvedores ← GetDesenvolvedoresSeguindo(login) ;  /* Busca
 os desenvolvedores que o usuário segue, utilizando
 a API do GitHub para realizar a ação */
foreach desenvolvedor ∈ desenvolvedores do
    repositorios ← BuscaRepositoriosDev(desenvolvedor.login);
     /* Busca as informações dos repositorios do
     desenvolvedor que está analisando */
    correspondenciareadme ←
     VerificaCorrespondenciaReadme(repositorios);  /* Verifica
     as correspondências nos readme dos repositórios
     */
    correspondencialinguagem ←
     VerificaCorrespondenciaLinguagem(repositorios);
     /* Verifica as correspondências das linguagens
     */
    correspondenciamensagenscommits ←
     VerificaCorrespondenciaCommits ;          /* Verifica a
     correspondência nas mensagens dos commits dos
     repositórios */
    correspondenciaDev ← correspondenciareadme +
     correspondencialinguagem + correspondenciacommits;
     /* Realiza a soma das correspondências para
     comparar com outros desenvolvedores */
    desenvolvedor.correspondencia ← correspondenciaDev
end
desenvolvedorCorrespondente ←
 RetornaDevMaiorCorrespondencia(desenvolvedores);
return desenvolvedorCorrespondente
```

O algoritmo 1 apresenta o pseudocódigo da funcionalidade de atribuição. Ele recebe como entrada o login do desenvolvedor que fez a requisição e o tópico que ele deseja pesquisar. A partir dessa entrada, o HelpBot verifica nos repositórios públicos dos desenvolvedores que o desenvolvedor que realizou a requisição segue, analisando as linguagem e o leia-me dos repositórios. A saída desse algoritmo é o desenvolvedor de

software que mais teve correspondência sobre o tópico pesquisado.

A funcionalidade de agregação que no robô é tratado como "Quais"ocorre quando o desenvolvedor quer pesquisar quais desenvolvedores que o seguem possuem conhecimento em determinado tópico. Para isso, o HelpBot faz uma verificação nos repositórios públicos dos desenvolvedores que seguem o desenvolvedor que fez a requisição. Para essa funcionalidade, o retorno do HelpBot é a lista de desenvolvedores ordenados em ordem decrescente de acordo com a correspondências com o tópico que foi feita a requisição.

Algorithm 2: Pseudcódigo funcionalidade de agregação do HelpBot

input : Requisição realizada para o robô passando os dados necessários em JSON - tópico da pesquisa
output: Lista com os logins dos desenvolvedores que tiveram correspondência com a pesquisa, em ordem decrescente

```
login ← douglaswcastro;      /* login do desenvolvedor que
 realização a requisição para o robô */
topico ← Python ; /* Tópico informado pelo desenvolvedor
 para realizar a pesquisa */
desenvolvedores ← GetDesenvolvedoresSeguidores(login) ; /* Busca
 os desenvolvedores que seguem o que realizou a
 requisição utilizando a API do GitHub para
 realizar a ação */
foreach desenvolvedor ∈ desenvolvedores do
    repositorios ← BuscaRepositoriosDev(desenvolvedor.login);
     /* Busca as informações dos repositorios do
     desenvolvedor que está analisando */
    correspondenciareadme ←
     VerificaCorrespondenciaReadme(repositorios);  /* Verifica
     as correspondências nos readme dos repositórios
     */
    correspondencialinguagem ←
     VerificaCorrespondenciaLinguagem(repositorios);
     /* Verifica as correspondências das linguagens
     */
    correspondenciamensagenscommits ←
     VerificaCorrespondenciaCommits ;          /* Verifica a
     correspondência nas mensagens dos commits dos
     repositórios */
    correspondenciaDev ← correspondenciareadme +
     correspondencialinguagem + correspondenciacommits;
     /* Realiza a soma das correspondências para
     comparar com outros desenvolvedores */
    desenvolvedor.correspondencia ← correspondenciaDev
end
desenvolvedoresCorrespondentes ←
 OrdenaDesenvolvedores(desenvolvedores);
return desenvolvedoresCorrespondente
```

O algoritmo 2 apresenta o pseudocódigo da funcionalidade de agregação. Ele recebe como entrada o login do desenvolvedor que fez a requisição e o tópico que ele deseja pesquisar. A partir dessa entrada, o HelpBot verifica nos repositórios públicos dos desenvolvedores que seguem o desenvolvedor que realizou a requisição, analisando as linguagens e o leia me dos repositórios. A saída desse algoritmo é a lista de desenvolvedores ordenadas em forma decrescente pela correspondência com o tópico.

4.1.2. Requisitos Não funcionais

1. O robô só terá acesso aos repositórios públicos dos desenvolvedores de *software*.
2. O robô deverá ser desenvolvido na linguagem *Python*.
3. O desenvolvedor de *software* precisa estar logado no *Github* para realizar o comentário no repositório do robô.

4.1.3. Cenários de Uso

Um cenário de uso do robô é quando um desenvolvedor que nunca trabalhou com *Python*, quer descobrir quem dos desenvolvedores que ele segue tem familiaridade com a linguagem. Ele pode realizar um comentário no ultimo *commit* do repositório do robô com a menção a ele ”HelpBot”, mencionando o Quem e o o tópico que ele deseja como a linguagem *Python*. A Figura 1 traz o exemplo do cenário acima, com a requisição realiza para o robô e a resposta que ele retornou para o desenvolvedor. A Figura 2 mostra a requisição da funcionalidade de atribuição que não teve correspondência na pesquisa.

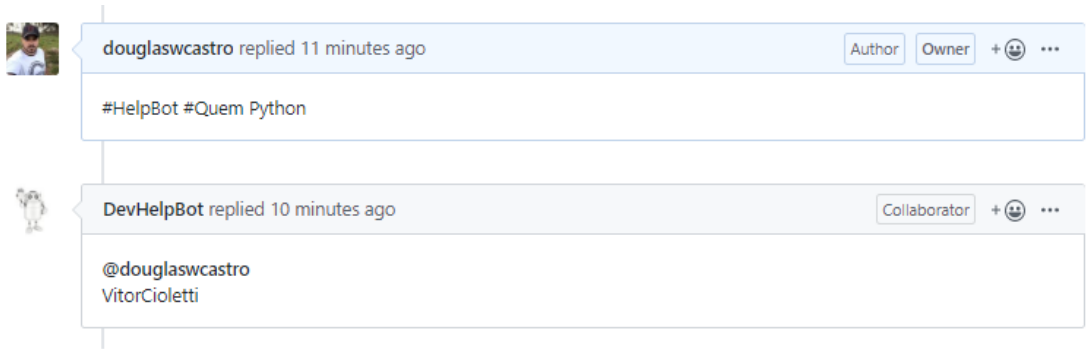

Figura 1. Exemplo da requisição e da resposta descrita no cenário da funcionalidade de atribuição

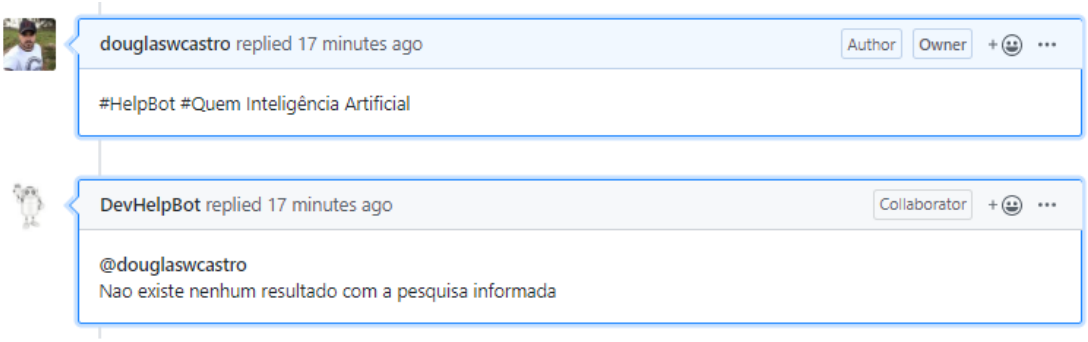

Figura 2. Exemplo da requisição e da resposta descrita no cenário da funcionalidade de atribuição, quando a pesquisa não encontra alguma correspondência.

Outro cenário é um desenvolvedor querer saber, dos seus seguidores, aqueles que já tiveram algum experiência com HTML. O desenvolvedor pode comentar com a citação do robô HelpBot, mencionando a funcionalidade ”Quais”e o tópico ”HTML”. Com isso, o robô realiza as requisições a *API* do *GitHub* e retorna os dados processados para o usuário. A Figura 3 mostra o exemplo do cenário descrito acima. A Figura 4 mostra a requisição da funcionalidade de agregação que não teve correspondência na pesquisa. Quando isso acontece o retorno é uma mensagem de que não ocorreu nenhum resultado com a pesquisa informada.

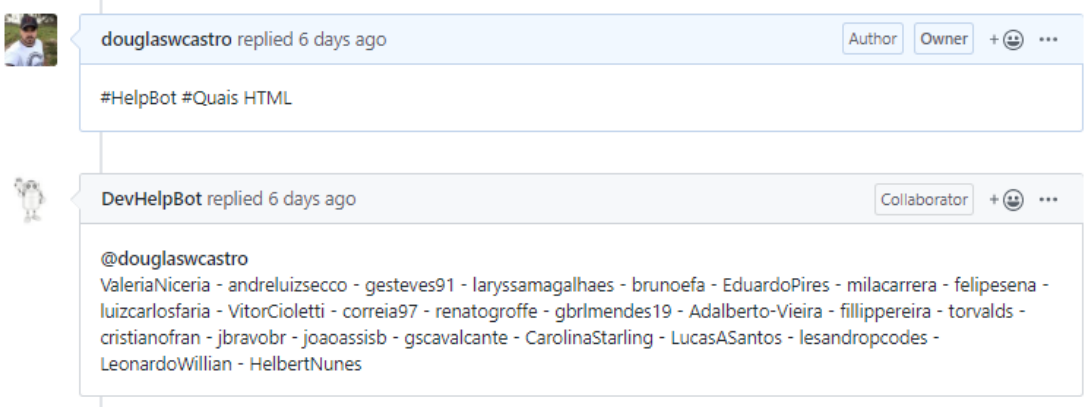

Figura 3. Exemplo da requisição e da resposta descrita no cenário da funcionalidade de Agregação

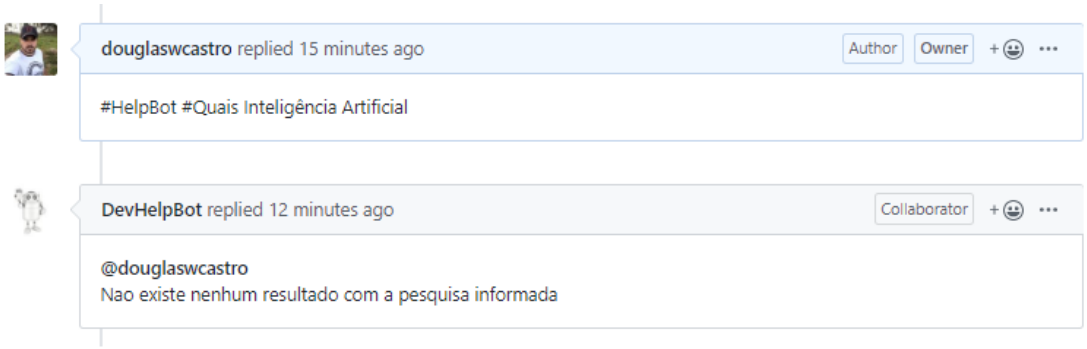

Figura 4. Exemplo da requisição e da resposta descrita no cenário da funcionalidade de agregação

4.2. Implementação

O robô foi desenvolvido em *Python* usando as bibliotecas *requests*, *json*, *os* e *flask*. Essa última foi usada para criar o *endpoint* chamado pelo *WebHooks*, um gatilho configurado no próprio *GitHub*, para que através de um evento (nesse caso, o comentário), ele dispare uma requisição para o domínio específico. A hospedagem do robô foi feita no *Heroku* (heroku.com), onde foi gerado um domínio fixo, usado para receber as requisições. *Heroku* é um serviço de hospedagem gratuito que pode ser configurado, uma vantagem é que ele é totalmente integrado com o *Github*, realizando *deploy* automático a cada *commit* na *branch master* do repositório.

O robô utiliza a *API* do *Github* para obter os dados necessários para a busca que o usuário solicitou, como os métodos de busca dos seguidores, os que seguem o usuário, busca os repositórios de usuários e o de criar um comentário no *commit* do repositório, para responder o usuário que fez a solitação. Citando o usuário e com a lista de usuários que apareceram na busca, se não existir nenhum resultado da busca, uma mensagem avisa ao usuário que não teve resultados. Através da marcação do usuário no comentário a própria plataforma do *GitHub* envia um *e-mail* para o usuário o avisando que ele foi citado em um comentário e com o link para acessar diretamente o comentário que foi citado.

A Figura 5 mostra um diagrama de sequência que mostra a ordem das ações das interações com o robô, com o desenvolvedor realizando o comentário no *commit* e através do gatilho *WebHooks*, é feita a chamada para o robô passando um Notação de Objeto *JavaScript* (JSON, do inglês *JavaScript Object Notation*). Após o robô ser chamado ele processa os dados recebidos e realiza requisições a *Api* do *GitHub* para buscar os dados necessários para retornar a busca informada, assim que termina o processamento o robô realiza o comentário em resposta ao usuário por meio do próprio *GitHub*, avisando o usuário que ele foi citado em uma mensagem.

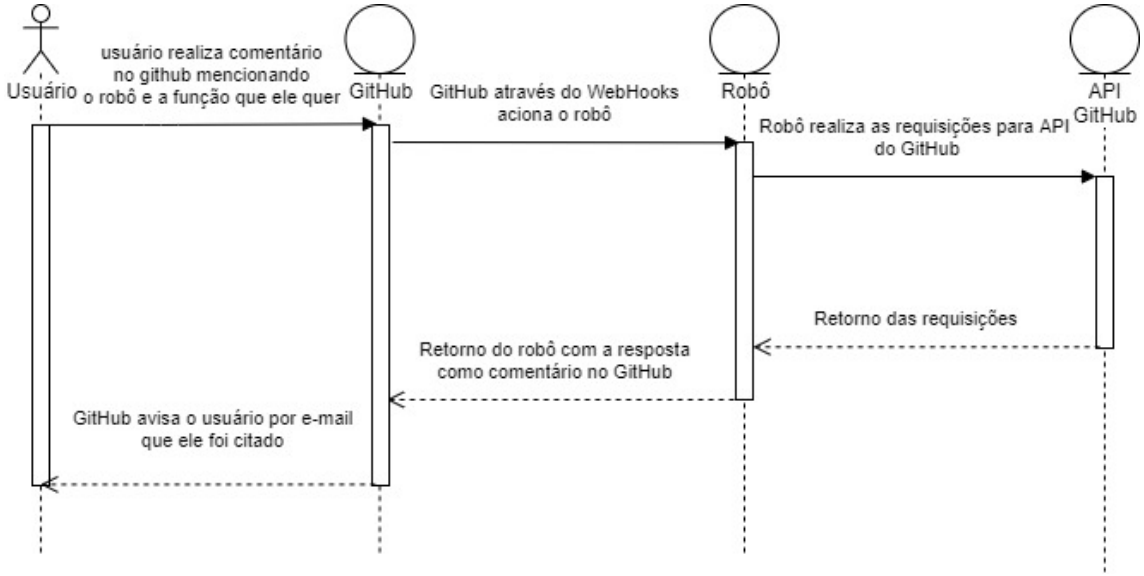

Figura 5. Diagrama de sequência exemplificando as interações para o uso do HelpBot

Link para o repositório no *GitHub*: https://github.com/douglaswcastro/HelpBot.

5. Avaliação do Robô

Nesta seção são tratadas as avaliações feitas no HelpBot, trazendo testes feitos por cinco desenvolvedores que, a partir da utilização do robô, responderam três perguntas sobre o HelpBot. Os resultados das avaliação serão apresentados e detalhados, junto com as discussões e implicações sobre o trabalho.

5.1. Metodologia: Testes funcionais

Tabela 1. Informações sobre os desenvolvedores que fizeram os testes no HelpBot

Desenvolvedor de Software	Tempo no *GitHub* em Meses	Seguidores	Seguindo	Repositórios
Desenvolvedor 1	41	12	31	17
Desenvolvedor 2	24	8	9	5
Desenvolvedor 3	24	5	8	2
Desenvolvedor 4	29	7	13	4
Desenvolvedor 5	24	2	40	2

Foram acompanhados os testes de quatro dos desenvolvedores e, a partir desse acompanhamento, foi notado que nenhum deles teve problemas para utilizar o *HelpBot*. Após as explicações das funcionalidades, os 4 desenvolvedores fizeram aquilo que era esperado e passaram os dados certos. Foi observado caso algum deles informasse parâmetros que não eram esperados pelo *HelpBot*, podendo gerar alguma falha. O desenvolvedor que não foi acompanhado informou que não teve dificuldade para utilizar o robô após a explicação que foi realizada para todos.

Após a análise realizada junto aos desenvolvedores, a partir do acompanhamento dos testes e pelo comentário do desenvolvedor que não foi acompanhado, os cinco desenvolvedores não tiveram problema para a utilização do *HelpBot* e avaliaram o uso de fácil entendimento e entenderam as funcionalidades. O acompanhamento foi realizado para entender a utilização dos desenvolvedores e se algum deles teria alguma dificuldade para o uso. Os desenvolvedores deram as suas opiniões sobre o *HelpBot* e, após a explicação, acharam a utilização bem intuitiva e simples, já que eles precisaram somente realizar um comentário passando os parâmetros corretos que o robô responde, marcando-o no comentário. Com isso, ele é avisado por *e-mail* assim que é respondido.

Os testes do HelpBot foram feitos por cinco desenvolvedores com interesses em tópicos diferentes. Antes de começarem os testes foi feita uma explicação do funcionamento do HelpBot para que entendessem as suas funcionalidades. Após isso, foram realizados os testes com as pesquisas dos tópicos que possuem interesses. Após os testes foram enviados para os desenvolvedores um formulário com as seguintes perguntas:

1. O quão útil você considera o HelpBot e as suas funcionalidades?
2. Você usaria o HelpBot?
3. Qual é a probabilidade de recomendar o HelpBot?

5.2. Resultados da Avaliação

Os casos de testes do robô são apresentados através da Tabela 2. Na coluna desenvolvedor de *software* é mostrado qual desenvolvedor fez a pesquisa no HelpBot. Na coluna de funcionalidade é informado a função que foi passada para o robô, ”Quem”é o parâmetro para a funcionalidade de atribuição e ”Quais”a funcionalidade de agregação. A coluna de pesquisa traz o termo utilizado para a busca. A última coluna traz o resultado da pesquisa que foi feita pelo robô.

Tabela 2. Requisições feitas para o HelpBot pelos desenvolvedores que o testaram, mostrando os dados que foram informados e os resultados das pesquisas

Desenvolvedor de Software	Funcionalidade	Pesquisa	Resultado
Desenvolvedor 1	Quem	Python	VitorCioletti
Desenvolvedor 2	Quais	Java	YitzhakAndrade - rcioletti - felipesena - douglaswcastro - EduardoPires
Desenvolvedor 3	Quais	Python	Não existe nenhum resultado para a pesquisa informada.
Desenvolvedor 4	Quais	C#	VitorCioletti - douglaswcastro - felipesena - devitcabelo - joaoassisb - fillippereira - HelbertNunes
Desenvolvedor 4	Quais	Python	VitorCioletti - gesteves91 - felipesena - harlleyaugusto - glucianog - douglaswcastro - laryssamagalhaes - fillippereira - HelbertNunes
Desenvolvedor 5	Quem	Python	douglaswcastro

O resultado da funcionalidade de agregação vem em ordem decrescente, ordenado pela correspondência com o tópico pesquisado nos repositórios dos desenvolvedores, já para a funcionalidade de atribuição o retorno é o desenvolvedor que teve maior correspondência com o tópico. A Tabela 2 mostra os parâmetros que foram passados para o robô pelos desenvolvedores e mostra a resposta para cada um dele, sendo que o desenvolvedor 4 realizou dois teste para a mesma funcionalidade.

Junto com os testes dos desenvolvedores foi verificado o tempo do resposta do HelpBot, avaliando o tempo que ele demorou para responder aos desenvolvedores a partir das pesquisas realizadas. O tempo máximo foi de 2 minutos com o mínimo de 15 segundos, a mediana do tempo de resposta é 1:10 minutos, a média foi de 1:09 minutos. Esse

número pode variar dependendo da quantidade de desenvolvedores que está na rede de amizade e pela quantidade de repositórios que eles possuem.

Os desenvolvedores que realizaram os testes no HelpBot escolheram tópicos do seu interesse para realizar a pesquisa no robô. Com isso, após o uso das funcionalidades que o HelpBot possui, eles receberam um formulário com as perguntas citadas na seção 5.1, para que fosse avaliado o HelpBot. Após as respostas dos cinco desenvolvedores, foi criada a tabela 3 para mostrar o resultado da avaliação. A tabela foi organizada com o número do desenvolvedor que respondeu a avaliação e com as suas respostas seguindo a ordem das perguntas mostradas na seção 5.1.

Tabela 3. Resposta do questionário aplicado aos desenvolvedores que realizaram os testes no HelpBot

Desenvolvedor de Software	Pergunta 1	Pergunta 2	Pergunta 3
Desenvolvedor 1	Útil	Sim	Moderadamente provável
Desenvolvedor 2	Muito útil	Sim	Muito provável
Desenvolvedor 3	Muito útil	Sim	Muito provável
Desenvolvedor 4	Útil	Sim	Muito provável
Desenvolvedor 5	Útil	Sim	Muito provável

Conforme mostrado na Tabela 3, as avaliações dos cinco desenvolvedores foram positivas para o uso do robô. Com a pergunta 1, dois informaram que o *HelpBot* e as suas funcionalidades são muito úteis, com os outros três desenvolvedores informando que acharam útil. Na pergunta 2, todos os desenvolvedores informaram que caso precisassem de algum auxílio para o problema que o *HelpBot* ajuda a solucionar, o usariam. Já na pergunta 3, quatro desenvolvedores informaram que é muito provável que eles indiquem o *HelpBot* para algum desenvolvedor que eles saibam que necessita de tal ajuda. Um desenvolvedor respondeu que é moderavelmente provável que ele indique o robô.

5.3. Discussões e implicações

Neste trabalho foram abordados robôs na plataforma *GitHub*, onde atualmente poucos são encontrados. Com isso, este trabalho visou o desenvolvimento de um robô para tal. Apesar de ser desenvolvido com funcionalidades simples, o HelpBot foi concebido para ser um exemplo de robô de automação de tarefas no *GitHub*, podendo servir de exemplo e base para outros desenvolvedores que queiram desenvolver robôs para a mesma plataforma.

É um exemplo de um robô que recebe as requisições do *GitHub*, processando dados e retornando para o próprio *GitHub*, configurando requisições que são realizadas através da plataforma, juntamente de exemplo de bibliotecas que podem ser usadas caso os desenvolvedores queiram usar a linguagem de programação *Python*. Exemplos de requisições para a *API* do *GitHub* com exemplos práticos que podem ser usados de exemplos para outros desenvolvedores.

6. Conclusão

Este trabalho se propôs a desenvolvedor um robô para a plataforma *GitHub* com as funcionalidades de agregação e atribuição. Para alcançar o objetivo foi implementado o *Help-*

Bot com as duas funcionalidades, utilizando a linguagem de programação *Python* e a *API* do *GitHub* para coletar os dados necessários dos desenvolvedores na plataforma com o *HelpBot* recebendo a requisição, processando os dados e verificando as correspondências e respondendo a pesquisa pelo comentário do *GitHub*.

O objetivo do trabalho, que é o desenvolvimento do *HelpBot* para a plataforma *GitHub*, O robô foi desenvolvido e seu funcionamento validado como se observou na pesquisa se satisfação. Os resultados mostram que os cinco desenvolvedores responderam a pesquisa positivamente, tanto para as funcionalidades do robô quanto para a sua utilidade. Analisando as respostas do *HelpBot* para as pesquisas, a resposta foi retornada em no máximo dois minuto e com o tempo mínimo de 15 segundos.

Em relação a trabalhos futuros, podemos adicionar maiores funcionalidades ao *HelpBot*, tornando-o um robô de automação de repositórios, com funcionalidades mais complexas e que ampliem a sua atuação na plataforma, podendo ser usado para um melhor controle de repositórios. Adicionando funcionalidades de controle de *Pull Requests* e controle de versionamento, ajudando a controlar e verificar as mudanças em repositórios.

Referências

Beschastnikh, I., Lungu, M. F., and Zhuang, Y. (2017). Accelerating software engineering research adoption with analysis bots. In *Proceedings of the 39th International Conference on Software Engineering: New Ideas and Emerging Results Track*, ICSE-NIER '17, pages 35–38, Piscataway, NJ, USA. IEEE Press.

Braz, P. A. and Goldschmidt, R. R. (2018). Redes neurais convolucionais na detecção de bots sociais: Um método baseado na clusterização de mensagens textuais. In *SBSeg 2018*, pages 323–336, Porto Alegre, RS, Brasil. SBC.

D'Ambros, M., Gall, H., Lanza, M., and Pinzger, M. (2008). *Analysing Software Repositories to Understand Software Evolution*, pages 37–67. Springer Berlin Heidelberg, Berlin, Heidelberg.

Dunham, K. and Melnick, J. (2008). *Malicious bots: an inside look into the cybercriminal underground of the internet*. Auerbach Publications.

Fidalgo, R., Silva, E., and Franco, N. (2015). Classificando e implementando feedbacks para aprendizado ativo em ferramentas case: o caso eercase. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 26, page 308.

Freitas, C. A. S. (2014). Bots sociais: implicações na segurança e na credibilidade de serviços baseados no twitter. `http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm`. Acessado: 2019-05-23.

Hannebauer, C. and Gruhn, V. (2014). Algorithmic complexity of the truck factor calculation. In Jedlitschka, A., Kuvaja, P., Kuhrmann, M., Mannisto, T., Munch, J., and Raatikainen, M., editors, *Product-Focused Software Process Improvement*, pages 119–133, Cham. Springer International Publishing.

Jardim de Oliveira, H. and Ponciano, L. (2018). Conversa entre robôs: Implementando um robô assistente para mediar a interação com robôs no slack. Trabalho de Conclusão de Curso em Sistemas de Informação. PUC Minas. Unidade Barreiro.

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., and Damian, D. (2014). The promises and perils of mining github. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, MSR 2014, pages 92–101, New York, NY, USA. ACM.

Kollanyi, B. (2016). Automation, algorithms, and politics where do bots come from? an analysis of bot codes shared on github. *International Journal of Communication*, 10:20.

Lopes, A. and Garcia, G. (2002). Introdução à programação. *Editora Campus*, 6.

Nogueira, F., Ocana, K., Silva, V., Braganholo, V., and de Oliveira, D. (2017). Uma estratégia para versionamento dos dados de workflows científicos executados em nuvem.

Nunes, D. (2018). Inteligência artificial e direito processual: Vieses algorítmicos e os riscos de atribuição de função decisória às máquinas. *Revista de Processo— vol*, 285(2018):421–447.

Nunes, V. B. (2005). Integrando gerência de configuração de software, documentação e gerência de conhecimento em um ambiente de desenvolvimento de software. *Universidade Federal do Espírito Santo*.

Pereira, A. and Ponciano, L. (2019). Componentes de atribuição e agregação em sistemas de computação por humanos. In Arenare, F., Lobato, W., and Hanriot, S., editors, *Iniciação Científica: Destaques 2018*, pages 427–442. Editora PUC Minas, Belo Horizonte, MG, Brasil.

Sandim, H., Brandao, M. A., and Moro, M. M. (2018). Stf: uma abordagem social para estimar truck factor no github. `https://homepages.dcc.ufmg.br/~mirella/pdf/2018.VEM-Sandim.pdf`. Acessado: 2019-05-23.

Simão, J. and Nomura, S. (2017). Análise experimental de ferramentas case para documentação. `https://www.peteletricaufu.com/static/ceel/doc/artigos/artigos2014/ceel2014_artigo035_r01.pdf`. Acessado: 2019-05-22.

Yu, Y., Wang, H., Yin, G., and Wang, T. (2016). Reviewer recommendation for pull-requests in github. *Inf. Softw. Technol.*, 74(C):204–218.