Camila Naomi Kodaira - 4266602
Felipe Caetano Silva - 9293223

Projeto de Pesquisa
MsFLOSS: Um observatório de Comunidades de Software Livre

São Paulo - SP

2019

Camila Naomi Kodaira - 4266602
Felipe Caetano Silva - 9293223

Projeto de Pesquisa
MsFLOSS: Um observatório de Comunidades de Software Livre

Projeto de Pesquisa apresentado como Trabalho de Conclusão de Curso, como proposta para a atividade a ser desenvolvida ao decorrer do ano letivo.

Universidade de São Paulo - Instituto de Matemática e Estatística

Orientador: Paulo Meirelles
Coorientador: Fabio Kon

São Paulo - SP
2019

Lista de abreviaturas e siglas

MsFLOSS	Miss Free/Libre Open Source Software
FLOSS	Free Libre Open Source Software
GNU	GNU's Not Unix
GCC	GNU Compiler Collection
USP	Universidade de São Paulo
NFS	Network File System
Linux-IIO	Linux Industrial Input Output
FLUSP	FLOSS at USP
IME	Instituto de Matemática e Estatística
IRC	Internet Relay Chat

Sumário

1 Introdução

O desenvolvimento do projetos de software livre e a dinâmica de suas comunidade é tema de diversos estudos, entre os mais famosos está o de Raymond [**1**] que, em seu ensaio, define o desenvolvimento de software livre como um Bazar e declara a seguinte tese: *"Given enough eyeballs, all bugs are shallow"*, que ficou conhecida como "Lei de Linus". Como resultado desse ensaio, o desenvolvimento de software livre foi identificado como um fenômeno homogêneo e idealizado.

Um dos aspectos do desenvolvimento do software livre é o caráter voluntário de seus contribuidores, sendo muito elogiado por Raymond, pois para ele os contribuidores são nada mais que usuários especialistas do sistema e assim que encontram bugs, realizam esforços para reportar e até consertar por si mesmos. Embora que durante muitos anos a tese de Raymond tenha sido muito referenciada pela comunidade de engenharia de software, alguns estudos começaram a questionar alguns aspectos da dicotomia "Catedral e Bazar". Uma das discordâncias é com relação ao caráter homogêneo que o desenvolvimento de software livre tem sido retratado, ao utilizar como exemplo as comunidades que contribuem com os projetos Linux Kernel e GCC, sistemas bem sucedidos, muitos outros projetos de software livre são vistos como idênticos como cita Osterlie [**2**]. Outro aspecto divergente do desenvolvimento do software livre é com relação a motivação dos seus contribuidores, uma vez que empresas consideradas grandes "players" no mundo comercial se interessam pelo desenvolvimento com código aberto, porém com certas restrições, isto é, criando versões pagas do sistema para que o usuário usufrua do produto de forma completa, fato esse analisado por Fitzgerald [**3**].

Nesse contexto se torna muito importante observar a dinâmica das comunidades de software livre atual, para que desta forma seja possível reunir o que é visto na prática deste desenvolvimento com a teoria e então preencher a lacuna entre os dois. A plataforma MsFLOSS apresenta-se como uma ferramenta fundamental para a transição de um modelo teórico onde a pesquisa gera metodologias que não são usadas na indústria para um modelo de laboratório, onde as práticas na indústria são observadas.

2 Objetivo

O principal objetivo do MsFLOSS é coletar dados sobre o desenvolvimento de projetos de software livre, para, com eles, verificar se a proposta de Raymond, apresentada há mais de 20 anos atrás, é a realidade do mundo atual. O MsFLOSS será uma plataforma para minerar dados dos meios de comunicação públicos mais utilizados pelas comunidades de software livre: IRC, listas de email, commits e issues; e então apresentá-los de forma que fique clara o impacto que cada membro da comunidade tem no desenvolvimento do software estudado.

Depois de criado o MsFLOSS, o objetivo do trabalho é utilizar essa ferramenta para analisar padrões nas comunidades de software livre, com um enfoque especial na influência de membros que trabalham no projeto como empregados de uma empresa. Queremos observar o quão chave são certas pessoas e empresas no progresso de um software livre, e com essa informação possivelmente desmistificar o software livre como uma única forma de trabalho utópica.

MsFLOSS será projeto de software livre que observa o desenvolvimento de outros projetos de software livre. A ideia é que ele possa ser uma ferramenta de análise da interações nas comunidades de software livre, para que os membros das respectivas comunidades possam verificar o quão dependentes eles são de certas pessoas, e com isso, poderem se reestruturar para que essa corrente de contribuição não seja tão frágil.

3 Metodologia de desenvolvimento e andamento do projeto

O projeto MsFLOSS é parte de uma pesquisa do IME-USP, no contexto do Grupo de Sistema, coordenado pelo Prof. Fabio Kon. Por ser um projeto grande, um protótipo também está sendo desenvolvido da matéria de MAC0413/5714, Programação Orientada a Objetos, ministrada pelo Prof. Fabio Kon, com a prioridade de prototipar quatro microserviços que irão minerar os dados. O projeto foi concebido a partir das pesquisas lideradas pela mestranda Melissa Wen, sob orientação do Prof. Paulo Meirelles. Em resumo, o desenvolvimento do projeto MsFLOSS nasceu de forma colaborativa.

O desenvolvimento do MsFLOSS seguirá práticas das Metodologias Ágeis de desenvolvimento de software e da Comunidades de Software Livre, com os autores deste trabalho de conclusão de curso trabalhando tanto como desenvolvedores, quanto como mantenedores do projeto, revisando as contribuições que serão aceitas no repositório oficial. Durante o primeiro semestre, o trabalho será colaborativo com os alunos de MAC0413/5714; já no segundo semestre, trabalharemos para evoluirmos a plataforma MsFLOSS, gerando uma versão para ser usado nas pesquisas do Grupo de Sistemas do IME-USP.

Na segunda fase de desenvolvimento do projeto MsFLOSS, com os microserviços coletando dados, planejamos que sejam disponibilizados para as comunidades de software livre visualizações de dados, para os desenvolvedores dos projetos poderem fazer análises sobre as interações que ocorrem em suas comunidades.

3.1 Arquitetura proposta

Nesta seção, apresentamos os componentes oferecidas pela plataforma, assim como uma arquitetura inicialmente proposta. Sendo uma arquitetura de microserviços, é vital que cada um do componentes listados abaixo mantenha independência das outras e, com isso, possam ser desenvolvidas de modo paralelo.

- A plataforma necessitará de robôs mineradores (microserviços) que precisarão reunir os dados que serão usados na análise, porém haverá uma forma de persistência, pois devem manter a sua independência dos outros módulos da plataforma. Serão inicialmente 4 comunidades a serem observadas: IRC, Listas de email, Commits no repositório Git e Issue Trackers como Bugzilla.
- A Vizualização de Dados também deverá ser um microserviço oferecido proporcionando gráficos, timelines, indicadores e núvens de palavras.

- Processamento de linguagem Natural será o módulo responsável por fazer análises de sentimento a partir dos dados gerados pelos robôs e fornecê-las ao módulo de visualização.
- O Repositório de Dados (ou Data colector) é o módulo responsável por organizar e agregar os dados recebidos dos serviços robôs e enviá-las ao módulo de visualização.
- A Integração será o módulo responsável por todas as atividades administrativas com relação aos usuários, autenticações, projetos, configurações, isto é, será o módulo responsável por integrar todos os microserviços.

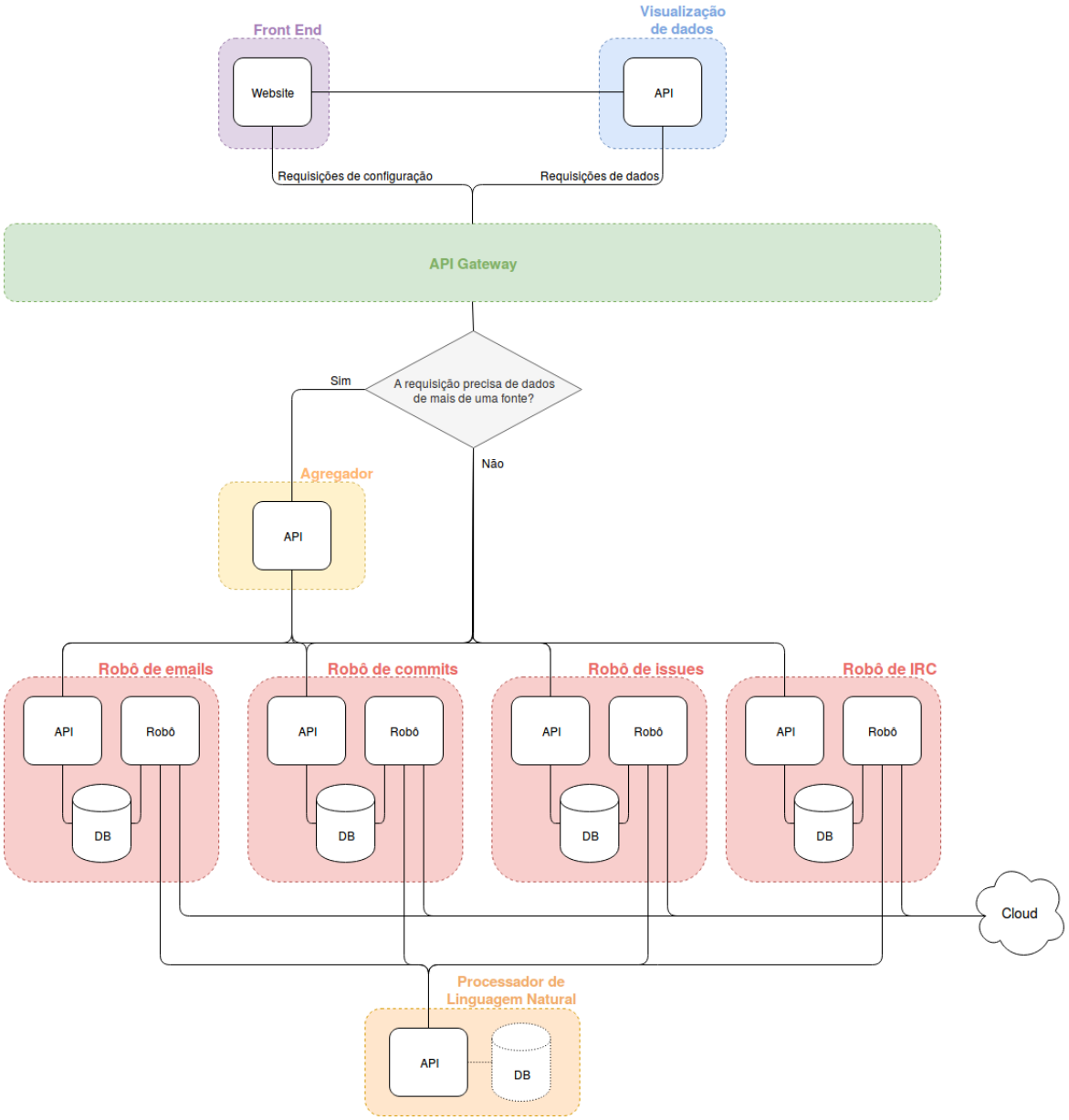

Figura 1 – Arquitetura do MsFLOSS

A relação entre esses microserviços está representada na Figura 1. Temos dois serviços para a representação gráfica do projeto, a aplicação (website) em si (representado

em roxo) e a API de visualização de dados (em azul). Ambos se comunicam com os outros serviços por meio de uma API Gateway (em verde), ou proxy; a aplicação fazendo requisições de configurações, como por exemplo enviando os endereços dos repositórios que o usuário quer monitorar, enquanto a visualização pede os dados para poder popular os gráficos requisitados pela aplicação.

As requisições podem ser atendidas de duas formas. Na primeira, ela precisa de dados apenas de um robô (em vermelho): nesse caso a própria API do robô irá responder. No segundo caso, se for necessário juntar dados recolhidos por dois ou mais robôs> nesse caso que o Agregador (em amarelo) é chamado. O Agregador é um código que junta dois datasets e repassa, não havendo um banco de dados especificamente para juntar toda a informação faz com que a plataforma seja escalável e gera desacoplamento entre os módulos.

Cada microserviço "minerador" possui duas partes: uma API que irá repassar os dados requisitados e o "robô" em si que irá minerar os dados. Todos os dados são armazenados em uma base de dados própria; essa base é acessada quando a API é requisitada, e é ela que retêm os dados de todo o serviço.

Quando necessário os microserviços chamam o Processador de Linguagem Natural (em laranja), que retorna os dados de sua análise. Esse serviço possui um banco de dados interno, que coleta informação dos robôs para armazenar contextos e fazer uma análise fina das discussões na comunidade. Entretanto, os dados desse banco são acessados apenas pelo processador; eles não são passados para os outros microserviços.

4 Cronograma

Nesta seção, apresentamos um cronograma das atividades previstas para este trabalho de conclusão de curso. Na Tabela 1, organizamos o cronograma de atividades que serão desenvolvidas ao longo do ano, separadas pela tarefa e pelo mês em que ela será trabalhada. O símbolo "*" representa o período de desenvolvimento colaborativo, referenciado na Seção 3, durante primeira fase do projeto, e o símbolo "x" as atividades que serão feitas normalmente, incluindo as atividades previstas para a segunda fase, no segundo semestre.

Mês/Atividade	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov
Proposta	x							
Monografia		x	x	x	x	x	x	x
Análise de Arquitetura	x	x						
Desenvolvimento de Robôs	*	*	*	x	x	x	x	
Análise dos dados				x	x			
Desenvolvimento da interface	*	*	*	x	x	x	x	
Refatoração de código				x				x

Tabela 1 – Cronograma de atividades

Descrição das atividades:

- Proposta: Escrita e revisão da proposta, inclui as alterações após sugestões do nosso orientador.
- Monografia: Escrita e revisão da monografia.
- Análise de Arquitetura: Revisão de arquiteturas existentes e decisão de qual usar.
- Desenvolvimento de Robôs: Escrita do código que irá coletar os dados.
- Análise de dados: Análise de dados obtidos e se eles são suficientes para atacar a questão problemática do nosso projeto.
- Desenvolvimento da interface: Escrita do código que irá apresentar os dados.
- Refatoração de código: Modificação e melhoramento da estrutura interna do código.

Bibliografia

[1] Eric Steven Raymond *The Cathedral and the Bazaar*, 1997.

[2] Thomas Østerlie and Letizia Jaccheri *A Critical Review of Software Engineering Research on Open Source Software Development*

[3] Brian Fitzgerald *The Transformation of Open Source Software*, 2006.