Universidade de São Paulo
Escola Politécnica
Departamento de Engenharia de Computação e Sistemas Digitais

Desenvolvimento de uma plataforma citizen science para coleta de dados em biodiversidade

Relatório de Atividades

Natanael Magalhães Cardoso
Orientador: Prof. Dr. Antônio Mauro Saraiva

Resumo

Citizen Science permite que cidadãos possam usar seu potencial de observação para contribuir em diversas áreas da ciência, desde a observação de um inseto até a classificação morfológica de uma galáxia. Mas, para que estas informações sejam utilizáveis em projetos científicos, é necessário que haja uma confiabilidade dos dados. Por isso, uma plataforma digital nesta área tem grande impacto, pois, além de organizar e agilizar a coleta e visualização dos dados, permite a implantação de um processo automático e eficaz de análise da qualidade dos dados. Projetos *citizen scrience* caracterizam-se pela troca de conhecimento entre os pesquisadores e os voluntários. Assim sendo, outra contribuição importante de uma plataforma digital é que a disposição dos envolvidos numa rede social facilita o engajamento dos cidadão na ciência e em outros projetos.

1 Introdução

Citizen science refere-se ao engajamento de voluntários amadores na investigação científica levantando questionamentos, coletando dados ou interpretando resultados. Em geral, projetos *citizen science* geralmente incluem uma parceria entre amadores e pesquisadores. [1] Num contexto de biodiversidade, os voluntários podem contribuir fazendo identificação de espécimes. Assim, é possível mapear e analisar as ocorrências de espécies em uma determinada região, identificando espécies exóticas ou em extinção, por exemplo. [2] Com o auxílio da tecnologia, é possível desenvolver plataformas que agilizem a coleta e o processamento dos dados. [3]

Uma plataforma online com estrutura de rede social que integra voluntários e pesquisadores é decisivo para o engajamento entre as pessoas

envolvidas e, consequentemente, a popularização do projeto. Os voluntários publicam suas observações e, ao mesmo tempo, podem aprender as pesquisas desenvolvidas. Esta troca de conhecimento entre profissionais e amadores é uma das principais características de projetos *citizen science*. [1]

Como os dados coletados por voluntários podem chegar a um grau de confiabilidade muito próximo ao de profissionais [4, 5], uma base de dados com observações georreferenciadas de espécimes coletadas por voluntários é uma riquíssima fonte de dados para projetos de pesquisa da área. Mais do que isso, uma plataforma que conecta pesquisadores e voluntários tem grande potencial de intercâmbio de informações.

2 Objetivos

Desenvolver uma plataforma digital de *citizen science* em biodiversidade que auxilie na coleta e divulgação de observações da fauna da Cidade Universitária Armando de Salles Oliveira. E, além disso, promover uma maior interação entre voluntários e pesquisadores, criando uma comunidade virtual de pessoas engajadas na observação e conservação dos bichos do campus.

É muito importante, também, garantir uma alta qualidade e confiabilidade nos dados adquiridos pelos voluntários para que possam ser utilizados em pesquisas da área. Por este motivo, a implementação será realizada em duas etapas, primeiramente dentro e depois fora do campus.

A plataforma digital não será implementada do zero. Ao invés disso, será escolhido algum sistema já existente para adaptar e dar continuidade no desenvolvimento. Muitos desses sistesmas são implementados em outro idioma e para outras finalidades. Por isso, é importante disponibilizar o conteúdo do sistema em português e de acordo com as necessidades deste

projeto.

E, como a interação entre a comunidade de voluntários é um dos focos do projeto, a implementação de novos recursos, como a *gamificação*[1], serão peças-chave para este projeto.

3 Materiais e Métodos

3.1 Escolha da plataforma

Primeiramente foi feito o estudo do estado da arte das tecnologias utilizadas em coleta de observações da natureza feitas por voluntários [6, 7]. Com isso, foi possível fazer um levantamento das melhores plataformas existentes hoje. A finalidade deste levantamento foi basear a implementação deste projeto em um já existente e, a partir dele, fazer modificações e otimizações para que atenda às necessidades locais.

Algumas das plataformas analisadas foram: o *iNaturalist*[2], desenvolvido pela Academia de Ciências da Califórnia, o *Map of Life*[3], desenvolvido pela Universidade Yale, e o *eBird*[4], desenvolvido pela Universidade Cornell. O iNaturalist foi escolhido como plataforma de desenvolvimento. A escolha foi feita analizando critérios como frequência de atualização do código, uso de software livre, sistema de controle de qualidade de dados utilizado e disponibilidade em vários tipos de dispositivos (desktop, móvel, etc.).

[1] Aplicação de elementos de jogos em contextos não relacionados aos jogos, como sistema de pontuação e recompensas.
[2] Site do projeto: http://inaturalist.org.
[3] Site do projeto: http://mol.org.
[4] Site do projeto: http://ebird.org.

3.2 Infraestrutura e implantação

Por se tratar do desenvolvimento e implantação de um sistema já existente, o estudo de recursos exigidos, como a pilha de software[5] utilizada, serão indispensáveis para garantir a reprodutibilidade do sistema nos servidores do Laboratório de Automação Agrícola – LAA.

Para hospedagem do website (*back-end*[6] e *front-end*[7]), foi usada uma máquina virtual em um servidor do LAA rodando o Ubuntu 14.04 como sistema operacional e com um endereço de IP público configurado. Além disso, a rede interna da USP foi utilizada para transportar o tráfego externo de dados ao sistema.

Além de configurar o ambiente de produção e implantar o sistema, foi necessário popular o banco de dados com informações iniciais necessárias para o funcionamento do *back-end*, como a árvore taxonômica das espécies.

3.3 Desenvolvimento

Por se tratar de sistemas em desenvolvimento paralelo, a sincronização do código entre os projetos é muito importante. Por isso, o GIT[8] foi usado para o gerenciamento do código através do serviço Github.com. Este é o mesmo serviço usado pela CalAcademy[9] para manter o projeto iNaturalist.

Para início do desenvolvimento, dois repositórios foram clonados: o

[5]Conjunto de subsistemas ou componentes de software necessários para criar uma plataforma completa. Isto inclui o sistema operacional, o sistema de banco de dados, o *web server*, a *framework*, dentre outros.

[6]Parte de um sistema de computador que não está diretamente acessível ao usuário final, tipicamente responsável pelo armazenamento e manipulação de dados.

[7]Parte de um sistema de computador em que o usuário final interage diretamente.

[8]Sistema de Controle de Versão – SCV, usado para gerenciamento e automação de códigos.

[9]Academia de Ciências da Califórnia.

principal[10], com o código-fonte do *back-end* e do *front-end*, e o do aplicativo Android[11]. A implementação desta nova versão do iNaturalist foi feita paralelamente com o *upstream*.

Como desenvolvimento desta instânica do iNaturalist foi feito em paralelo com o desenvolvimento do projeto original, o código deste projeto foi periodicamente sincronizado com o projeto original. Esta sincronização foi feita como mostra a figura 1.

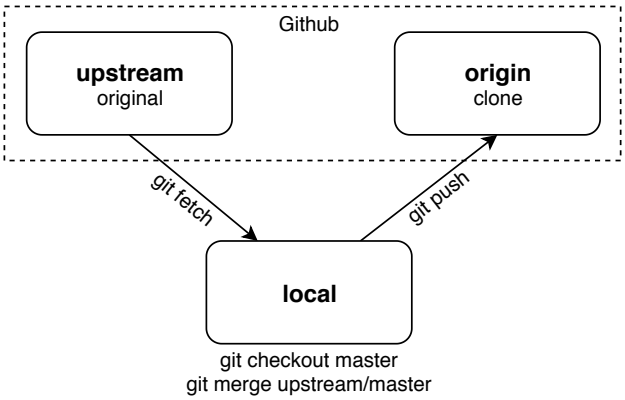

Figura 1: Diagrama da sincronização do código entre os repositórios.

Como o público álvo é brasileiro, a primeira tarefa foi traduzir todo aplicativo Android para o português. Depois, para incluir recursos de *gamificação* na plataforma, uma nova funcionalidade foi implementada: o sistema de *badges*. Para isto, foram necessárias algumas modificações na estrutura do banco de dados e no código.

[10]Endereço do repositório: https://github.com/inaturalist/inaturalist.
[11]Endereço do repositório: https://github.com/inaturalist/iNaturalistAndroid.

4 Resultados

Uma nova versão do iNaturalist, traduzida para o português e com resursos adicionais, foi implementada. Houve modificação em todos os principais componentes do sistema: website (front-end), servidor (back-end) e aplicativo Android.

Quando o desenvolvimento adquiriu um nível de estabilidade adequadod, o sistema foi implantado em uma máquina virtual do LAA com um IP fixo e com acesso à rede da USP, sendo, assim, disponibilizado para acesso de qualquer dispositivo pela internet. Esta infraestrutura implementada está ilustrada na figura 2.

O aplicativo Android não foi lançado na plataforma de distribuição de aplicativos oficial do Google, mas o arquivo de instalação foi disponibilizado no site para download e instalação manual.

5 Análises

5.1 Análise da Arquitetura

Nome	**Back-end**	**Android**
Linguagem de Programação	Ruby	Java
Framework	Ruby on Rails	Android SDK
Arquitetura de Software	MVC	–
Comunicação (API)[12]	RESTful (JSON)	RESTful(JSON)
Servidor HTTP	Apache	-
Banco de dados	PostgreSQL	SQLite

Tabela 1: Especificações do back-end e do aplicativo Android.

[12]Comunicação entre o *back-end* e o *front-end*.

O iNaturalist é um sistema complexo e com muitos recursos, que estendem desde o armazenamento dos dados, através das observações, até suas ferramentas de análises, pela sua interface web. A tabela 1 mostra as principais características, como as linguagens e arquiteturas utilizadas.

Tecnologias modernas e de código aberto foram muito bem exploradas no desenvolvimeto do sistema, como visto na tabela 1. *Ruby on Rails* é uma *framework* de código aberto, robusta e estável. É usada por grandes nomes como GitHub, Shopify, SoundCloud, Airbnb e outras.

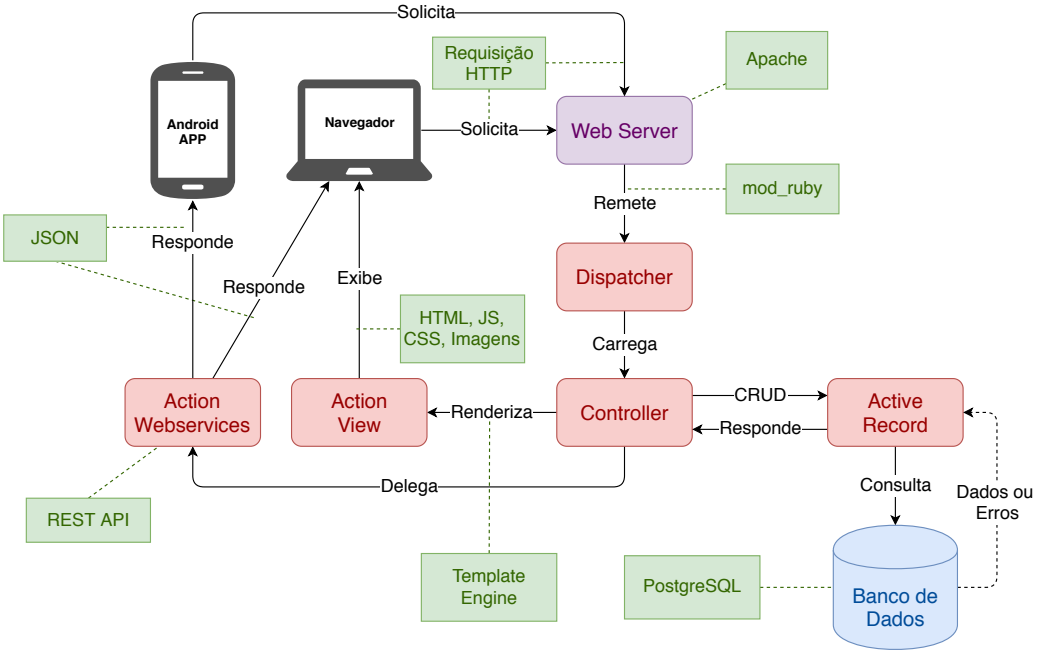

Figura 2: Diagrama simplificado da arquitetura do iNaturalist.

É notável, também, o uso arquitetura de software MVC (Model-View-Controler), que é a arquitetura padrão do *Ruby on Rails*. Ela divide a aplicação em três camadas de abstração interconectadas: modelo, visão e controlador. Ela é usada para separar representações internas de informações da representação final para o usuário do sistema. Isso organiza a criação de interfaces gráficas, uma vez que a camada de visualização está separada da

lógica interna do sistema.

A figura 2 mostra um diagrama simplificado da arquitetura do iNaturalist, nele é representado os componentes do sistema e as camadas de abstração da framework, mostrando como o *back-end* interage com o usuário dependendo do dispositivo usado para acessar os serviços.

Os blocos vermelhos são camadas de abstração do *Ruby on Rails*, que são representadas por classes. Os blocos verdes são informações adicionais de algum componente ou processo. E os demais blocos são componentes do sistema. Alguns dos elementos do diagrama serão descritos a seguir.

Web Server

Componente do sistema responsável por monitorar as requisições HTTP feitas ao servidor e encaminhar para a aplicação Ruby. Sua importância está no gerenciamento dos recursos do sistema, criando vários processos da aplicação para que seja possível executar requisições em paralelo.

Dispatcher

É uma das primeiras classes do fluxo de uma requisição. Ela que faz interface entre a rede e o sistema, recebendo a requisição HTTP do *Web Server* e instanciando o controlador da respectiva rota[13].

Controller

É implementada a lógica de uma ou de um conjunto de rotas identificadas por um padrão.

Active Record

Responsável por representar e manipular as tabelas do banco de dados.

[13]Neste contexto, rota está relacionada com o caminho completo da URL. Cada caminho tem um controlador associado, que pode renderizar uma página, fazer persistência de dados e etc.

Action View

É a classe de visualização e, portanto, é uma das últimas classe do fluxo de uma requisição. Ela recebe os dados do controlador manipula o template para renderizar uma página.

Action Webservices

É uma classe de apoio do *Ruby on Rails* para facilitar a criação de *Web Services* interoperáveis com a *framework*. Assim como a `Action View`, ela também está no final do fluxo de uma requisição, mas ao invés de renderizar uma página, ela processa e reponde requisições feitas pela API.

A arquitetura deste sistema é monolítica. Uma das principais características é que o sistema é segmentado por partes responsáveis por uma tarefa específica e estas partes são interconectadas. É possível observar isso analisando figura 2 e a descrição dos elementos.

Alguns pontos positivos da arquiterura monolítica são:

1. Desenvolvimento: no princípio, o desenvolvimento é muito simples, pois todo *codebase* está em uma mesma linguagem e em um mesmo ambiente.
2. Implantação: como todo código está em um mesmo ambiente, a implantação é simples: basta fazer uma cópia para o servidor.
3. Escalamento horizontal: para escalar o sistema, basta rodar múltiplas cópias atrás de um balanceador de carga.

Já alguns dos pontos negativos são:

1. Manuntenção: se a aplicação fica muito grande e complexa, fazer mudanças rapidatemente e corretamente torna-se um desafio, pois o encapsulamento de código é usado extensivamente pela arquitetura para separar o código em níveis de abstração, o que torna difícil fazer uma modificação sem gerar efeitos colaterais em outras partes do sistema.

2. Confiabilidade: um *bug* em qualquer módulo (ex: vazamento de memória) pode potencialmente derrubar todo processo. E, como todas as instâncias da aplicação são idênticas, este *bug* pode afetar o desenpenho de todo sistema.

3. Modernização: pode ser difícil novas tecnologias em aplicações monolíticas, já que modificações na linguagem ou na *framework* afeta toda a aplicação.

Com esses apontamentos, é certo concluir que a arquitetura adotada pelo projeto facilita a implantação da aplicação no servidor. Mas, por outro lado, existe um custo para fazer uma modificação no código, principalmente se a modificação envolver vários níveis de abstração.

Modificações como tradução ou qualquer outra na camada de visualização pode ser facilmente implementada, mas ao chegar em camadas mais baixas, como a lógica de negócio ou modelagem dos dados, é difícil fazer modificações sem romper a compatibilidade com o sistema original, que, por sua vez, também é modificado periodicamente.

5.2 Análise dos Novos Recursos

A possibilidade de fazer observações em um local mesmo que o smartphone não esteja conectado à internet é um ponto forte do aplicativo e garante que

as observações posam ser feitas de qualquer lugar. Os dados ficam armazenados na memória interna do aparelho e, quando conectado à rede, os dados são enviados para o servidor.

A partir das informações geográficas de latitude e longitude obtidas pelo próprio sensor do smartphone ou fornecidas pelo voluntário, assim que uma observação é enviada para o servidor, é adicionado um ponto no mapa representando a localização do espécime observado. Clicando sobre o ponto é possível obter informações detalhadas da observação. Além disso, para aprimorar a análise dos dados, é possível aplicar combinações de filtros que restringem a apresentação das observações no mapa. Essas restrições podem ser de acordo com o táxon, data ou local da observação. Assim, é possível observar a ocorrência de uma espécie em uma determinada época do ano ou sua presença em uma determinada região. Diversas combinações podem ser feitas de acordo com a necessidade da análise.

A estrutura de rede social cria vínculos entre os voluntários e pesquisadores e propicia o engajamento de mais voluntários à observação da natureza. Além disso, profissionais podem revisar a classificação do espécime feita pelos voluntários e sinalizar algo que esteja identificado incorretamente.

6 Conclusão

Na era dos grandes volumes de dados, sistemas capazes de organizá-los em informação efetiva.

Referências

[1] Jonathan Silvertown. "A new dawn for citizen science". Em: *Trends in Ecology & Evolution* 24.9 (2009), pp. 467–471. ISSN: 0169-5347. DOI: https://doi.org/10.1016/j.tree.2009.03.017. URL: http://www.sciencedirect.com/science/article/pii/S016953470900175X.

[2] Abraham Miller-Rushing, Richard Primack e Rick Bonney. "The history of public participation in ecological research". Em: *Frontiers in Ecology and the Environment* 10.6 (2012), pp. 285–290. DOI: 10.1890/110278. eprint: https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.1890/110278. URL: https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1890/110278.

[3] Rick Bonney et al. "Next Steps for Citizen Science". Em: *Science* 343.6178 (2014), pp. 1436–1437. ISSN: 0036-8075. DOI: 10.1126/science.1251554. eprint: https://science.sciencemag.org/content/343/6178/1436.full.pdf. URL: https://science.sciencemag.org/content/343/6178/1436.

[4] John Gollan et al. "Can Volunteers Collect Data that are Comparable to Professional Scientists? A Study of Variables Used in Monitoring the Outcomes of Ecosystem Rehabilitation". Em: *Environmental Management* 50.5 (nov. de 2012), pp. 969–978. DOI: 10.1007/s00267-012-9924-4. URL: https://doi.org/10.1007/s00267-012-9924-4.

[5] Arco J. van Strien, Chris A.M. van Swaay e Tim Termaat. "Opportunistic citizen science data of animal species produce reliable estimates of distribution trends if analysed with occupancy models". Em: *Journal of Applied Ecology* 50.6 (2013), pp. 1450–1458. DOI: 10.1111/1365-2664.12158. eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/1365-2664.12158. URL: https://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/1365-2664.12158.

[6] Azavea e SciStarter. *Citizen Science Data Factory. Part 1: Data Collection Platform Evaluation*. URL: http://www.azavea.com/index.php/download_file/view/1368/ (acesso em 15/12/2015).

[7] Azavea e SciStarter. *Citizen Science Data Factory. Part 2: Technology Evaluation*. URL: http://www.azavea.com/index.php/download_file/view/1369/ (acesso em 15/12/2015).