PROTOTIPAÇÃO DE APLICATIVO COM FUNCIONALIDADES DE SISTEMAS DE INFORMAÇÃO GEOGRÁFICA

André Sirikaku Miyashiro[1], Thiago Luís Lopes Siqueira[2]

[1] Graduando em Engenharia de Computação, Bolsista PIBIFSP, IFSP, Câmpus Piracicaba, a.sirikaku@aluno.ifsp.edu.br
[2] Orientador, Câmpus Piracicaba, prof.thiago@ifsp.edu.br
Área de conhecimento (Tabela CNPq): 1.03.03.04-9 Sistemas de Informação

RESUMO: Este artigo descreve a prototipação de um aplicativo com funcionalidades de sistema de informação geográfica (SIG) para dispositivos móveis. A motivação foi verificar a viabilidade técnica do projeto e desenvolvimento de um protótipo, isto é, descobrir se a tecnologia necessária existe e se pode ser empregada, e se o conhecimento técnico é suficiente. O desenvolvimento seguiu o Paradigma de Prototipação, foi utilizada a linguagem de programação Python junto de outros recursos compatíveis e o gerenciador de banco de dados MongoDB. Como resultado, o protótipo desenvolvido possibilita que o usuário visualize mapas, e neles dados geográficos representados por pontos e polígonos. Além disso, o protótipo do aplicativo permite que o usuário consulte o banco de dados conforme diferentes critérios e mostre nos mapas os resultados das suas consultas. Concluiu-se que as tecnologias e ferramentas existentes e utilizadas viabilizaram o desenvolvimento do protótipo de aplicativo com funcionalidades de SIG móvel.

PALAVRAS-CHAVE: SIG; Python; MongoDB; protótipo; viabilidade técnica.

PROTOTYPING AN APP WITH GEOGRAPHICAL INFORMATION SYSTEMS FUNCTIONALITIES

ABSTRACT: In this paper, we describe the prototyping of an app with functionalities that are intrinsic of mobile geographic information systems (mobile GIS). Our motivation was to verify the technical feasibility of the design and development of an app prototype, by investigating if the necessary technology exists and can be adopted, and whether the technical knowledge is sufficient. The software development followed the Prototyping Paradigm. We used the Python programming language together with other compatible libraries and the MongoDB database. As a result, the app prototype we developed enables users to visualize maps and geographical data represented as points and polygons on them. In addition, with the app prototype, users can query the database according to different criteria and display the query results on the maps. We concluded that the existing tools and technologies we used allowed the development of the app prototype with functionalities of mobile GIS.

KEYWORDS: GIS; Python; MongoDB; prototype; technical feasibility.

INTRODUÇÃO

Um Sistema de Informação Geográfica (SIG) consiste num sistema de informação projetado especialmente para armazenar, analisar e manipular dados geográficos. Ou seja, dados que representam objetos e fenômenos em que a localização geográfica é uma característica essencial para tratá-los (CÂMARA et al. 1996). As tecnologias emergentes para dispositivos móveis mudaram completamente como nos comunicamos e interagimos com o mundo exterior. Considerando o aumento desses dispositivos e o avanço das tecnologias de comunicação e informação, o SIG móvel tem viabilizado coletar, atualizar e consultar dados geográficos em tempo real. Sua plataforma integra

hardware, *software* e conectividade sem fio aos serviços. Deste modo, facilitam e tornam mais conveniente o acesso de usuários aos SIGs (HUANG, 2019; SUPERGEO, 2019). Além disso, o SIG móvel permite ampliar o uso do SIG para ambientes exteriores, além do uso tradicional *indoor*.

Estudos de viabilidade técnica comumente consideram se a solução ou tecnologia é prática (e sua maturidade), se a tecnologia necessária existe, e se o conhecimento técnico é suficiente. Uma das motivações do projeto cujas contribuições parciais são descritas neste artigo, foi realizar um estudo de viabilidade técnica para o desenvolvimento de um protótipo de SIG móvel. Huang (2019) enumera, dentre outros, os componentes seguintes de uma arquitetura para SIG móvel: dados (banco de dados geográfico e arquivos), mapas (servidores e formatos de dados intercambiáveis), e cliente móvel (edição de dados, serviços, consultas e armazenamento local). Aplicativos existentes dispõem desses recursos (SUPERGEO, 2019). Neste artigo, são descritos materiais e métodos usados na prototipação de um SIG móvel, bem como resultados alcançados, considerando-se os componentes enumerados, as tecnologias existentes e a (curva de) aprendizagem do bolsista (profissional em formação).

MATERIAL E MÉTODOS

O projeto do *software* seguiu o Paradigma de Prototipação (PRESSMAN; MAXIM, 2016), isto é, as etapas foram realizadas de forma iterativa. Primeiro, a obtenção de requisitos. Segundo a elaboração de projeto rápido. Terceiro, a construção de protótipo. Quarto, a avaliação de protótipo. Quinto, o refinamento do protótipo. Então, a primeira tarefa era reiniciada.

Utilizou-se a linguagem de programação *Python* 3.7 (HETTINGER, R. et al, 2020) cujas principais características são: permite orientação a objetos, é uma linguagem interpretada, é uma linguagem de programação interativa e possui uma sintaxe clara (HOLDEN, 2018).

Outra ferramenta utilizada para *Python* foi *Kivy*, uma biblioteca *open-source* para o desenvolvimento de aplicações multiplataforma (KIVY, 2020), dotada também de uma linguagem para organização da estrutura da interface, a *Kivy Language*. Essa linguagem permite a criação de uma interface moderna e rápida para as aplicações desenvolvidas usando Python e utilizadas por computadores pessoais ou dispositivos móveis (tais como smartphones).

Para viabilizar o componente de dados (banco de dados geográfico e arquivos), foi utilizado *MongoDB*, um gerenciador de banco de dados orientado a documentos (MONGODB, 2020). Seus dados são organizados em coleções contendo documentos estruturados no formato JSON (*JavaScript Object Notation*) (ECMA, 2017). Tal formato de dados padronizado armazena grupos de pares de chave-valor (MONGODB, 2020).

Foram criados os seguintes bancos de dados no *MongoDB*:

- *estabelecimentos* é uma coleção contendo 25462 documentos, cada documento contendo uma geometria do tipo ponto (*Point*), e disponibilizada pelo próprio *MongoDB*[1];
- *líneas* é uma coleção contendo 1000 documentos, cada documento contendo uma geometria do tipo linha (*LineString*), e tais linhas foram geradas artificialmente por um algoritmo[2];
- *bairros* é uma coleção com 5 documentos, cada um contendo um polígono (*Polygon*);
- *regiões* é uma coleção contendo 3759 documentos, cada documento contendo uma geometria do tipo polígono (*Polygon*), sendo que esta coleção foi obtida a partir de dados reais do censo.

Para acessar o banco de dados mantido no *MongoDB* usando-se *Python*, foi preciso o uso da biblioteca *pymongo*. Esta permite a conexão com o servidor e manipular e visualizar os documentos dentro das coleções (PYMONGO, 2020). Também foi utilizado um pacote para *Kivy*, chamado *MapView* que possibilita adicionar um mapa à interface e manipulá-lo, fazer desenhos nele ou adicionar marcadores em pontos específicos (LARKIN, 2019). GeoJSON foi o formato de dados intercambiável adotado, pois provê suporte a diferentes tipos de dados geográficos e suas representações com geometrias (BUTLER et al., 2016). *MapView* e GeoJSON foram empregados para viabilizar o componente de mapas (servidores e formatos de dados intercambiáveis). Durante o desenvolvimento foi usada a plataforma de controle de versões GitHub (GITHUB, 2020) para manter seguras e consistentes as cópias e versões do código-fonte.

[1] https://raw.githubusercontent.com/mongodb/docs-assets/geospatial/restaurants.json
[2] https://github.com/kivy-garden/mapview/issues/4

RESULTADOS E DISCUSSÃO

Um dos resultados obtidos se refere à implementação de um protótipo de aplicativo. A prototipação contribuiu significativamente com o trabalho, pois a partir da implementação foi possível conhecer a viabilidade técnica do desenvolvimento de cada componente de um SIG móvel. O protótipo é descrito conforme segue. Inicialmente, o protótipo exibe uma tela contendo um menu de opções, e sete delas são descritas a seguir.

Se o usuário selecionar a primeira opção, “Buscar por tipo”, uma nova tela é exibida. Nela, é mostrado um mapa. Uma barra localiza-se na parte superior, contendo o título da tela e um botão que leva para o menu. Essa barra se encontra presente em todas as outras telas, exceto no menu principal. Na parte inferior da janela do aplicativo existem duas caixas de texto: uma para digitar os tipos de comida que o usuário deseja procurar, e a outra para digitar o bairro que deseja fazer a busca. Existe também um botão com rótulo “Buscar” para o usuário pressionar e realizar a busca.

Após o botão de busca ser clicado, o programa irá tomar essas duas informações e fazer a consulta, no banco de dados, pelos estabelecimentos que possuem os tipos de comida informados e pelo bairro. São consultadas as coleções *estabelecimentos* e *bairros*. Concluída a consulta, a aplicação irá exibir pinos sobre o mapa, que representam os estabelecimentos encontrados, e desenhar um polígono com o contorno do bairro. Nesta opção, os componentes para armazenar, consultar e manipular dados geográficos são realçados. O mesmo acontece nas duas opções seguintes.

Caso escolha a segunda opção, “Adicionar estabelecimento”, o usuário é levado para uma tela com 4 caixas de texto para serem digitados os dados do item que se deseja adicionar no banco de dados e um botão escrito “Adicionar”. Junto com essas caixas há um texto acompanhado dizendo qual é a informação que se deve colocar naquele campo. Os dados que usuário deve colocar estão listados da seguinte maneira, de cima para baixo: nome do estabelecimento, os tipos de comida que ele serve, o preço médio de suas refeições, e por fim as suas coordenadas geográficas. Após pressionar o botão, o novo estabelecimento estará adicionado no banco de dados, especificamente na coleção *estabelecimentos*.

Caso escolha a terceira opção, “Deletar estabelecimento”, a tela que aparece para o usuário possui apenas uma caixa de texto e um botão escrito "Deletar”. Nela deve-se colocar o nome do estabelecimento que se deseja eliminar do banco de dados, e pressionar o botão. Será feita a busca pelo primeiro estabelecimento que corresponde, e será deletado.

Caso escolha a quarta opção, “Atualizar dados”, a tela irá permitir que o usuário altere os dados de um estabelecimento, cujo nome deve ser digitado. O usuário deve digitar também o nome do atributo (e.g. preço) que deseja alterar e o valor a ser associado (e.g. 49.99). Além dos três campos de texto citados, existe um botão cujo rótulo é “Atualizar”. Ao acionar o botão, o programa atualiza o estabelecimento correspondente no banco de dados. Caso o usuário forneça um valor que não corresponda com o atributo, por exemplo, escrever uma palavra ao invés de um número para “preço”, o programa irá mostrar um *popup* com uma mensagem de erro, pedindo para escrever um valor válido. Caso contrário, será feita a substituição do dado antigo pelo novo. Nesta opção, destaca-se o componente de atualizar dados geográficos.

Caso escolha a quinta opção, “Resumo do bairro”, o usuário é levado para uma tela com um mapa, idêntico às primeira e segunda opções. Nela, há uma caixa de texto para ser digitado o nome do bairro sobre o qual se deseja saber as informações dos estabelecimentos, e um botão rotulado “Busca”. Após clicar nele, será feita uma consulta no banco de dados e desenhado no mapa o contorno do bairro e um pino. Ao ser pressionado, o pino mostra outro botão com o texto “Menu”. Clicando nele, um *popup* abrirá e listará a quantidade de estabelecimentos, a média de preços dos estabelecimentos daquele bairro, e uma lista com os nomes deles. Para essa opção, a consulta no banco de dados foi diferente das outras opções, pois utilizou o operador *aggregate* e a função AVG. Nesta opção, destaca-se o componente de consultar dados geográficos.

Caso escolha a sexta opção, “Adicionar novo campo”, o aplicativo leva o usuário para outra tela, na qual permite ao usuário acrescentar um atributo novo ao estabelecimento e atribuir seu valor. Por exemplo, um atributo para avaliação e 8.5 como valor atribuído, respectivamente. É preciso informar o nome do estabelecimento, o nome do novo atributo e o novo valor dele. Depois de digitar, deve-se pressionar “Atualizar”. Então, será feita uma modificação no documento do estabelecimento

correspondente no banco de dados. Nesta opção, destaca-se também o componente de atualizar dados geográficos.

Caso escolha a sétima opção, “Consultar e exibir linhas e polígonos”, uma tela com mapa aparecerá, e junto dela uma barra com um botão rotulado “Desenhar”, um *slider* com intervalo entre 1 e 1000 com saltos de 1 unidade, um *label* mostrando o valor selecionado pelo usuário no *slider* e mais dois botões do tipo *ToogleButton* com rótulos ‘Linha’ e ‘Polígono’. Se houver clique em ‘Linha’ ou ‘Polígono’, o botão mudará para cor azul, sinalizando que o botão está selecionado. Ao clicar em “Desenhar”, será consultado o banco de dados, recuperada a quantidade de geometrias conforme indicado pelo usuário no *slider*, e mostrada no mapa uma quantidade de linhas ou polígonos dependendo do que foi selecionado anteriormente. Essa funcionalidade permite consultar o banco de dados e recuperar dele diferentes quantidades de geometrias, conforme tipos distintos. Caso nenhum dos botões for selecionado, ou ambos o forem, aparecerá um *popup* com uma mensagem de erro, pedindo para o usuário selecionar no mínimo uma opção. As coleções consultadas são *líneas* e *regiões*. Nesta opção, destaca-se também o componente de consultar dados geográficos.

Testes preliminares foram realizados no protótipo para verificar se a consulta e a exibição de números crescentes de geometrias iriam causar lentidão no processamento. Para tal, foram consultadas e exibidas 100, 500 e 1000 geometrias. Observou-se que a exibição de quantidades crescentes de pontos causa lentidão nas operações de *zoom* e deslocamento. Contudo, tal lentidão não foi percebida na exibição de linhas e polígonos. Nesses testes, contudo, notou-se que o tempo para o carregamento do mapa aumentou.

Como forma de ilustrar tanto a interface do aplicativo quanto a complexidade do processamento omitida por essa interface, a Figura 1 exibe a tela desenvolvida para a opção, “Resumo do bairro”. Na tela mostrada, o usuário deve: (1) digitar o nome do bairro que procura; (2) clicar no botão rotulado ‘Buscar’; (3) pressionar o cursor em cima do pino vermelho, abrindo um novo botão ‘menu’; e (4) clicar no botão e visualizar um *popup* que aparecerá contendo as informações do bairro em relação aos estabelecimentos dentro dele.

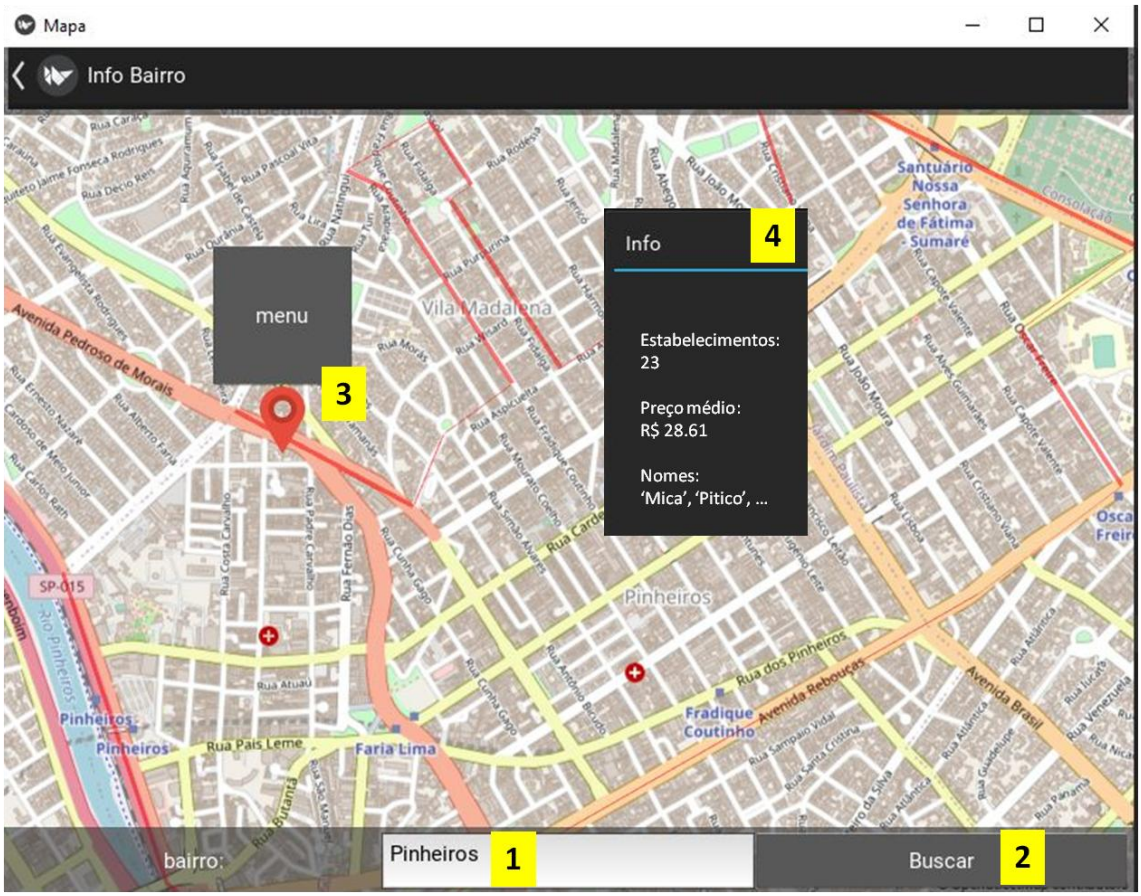

FIGURA 1. Ilustrando a funcionalidade do “Resumo do bairro”.

CONCLUSÕES

Neste artigo, descreveu-se o desenvolvimento de um protótipo de SIG móvel, a partir da motivação de verificar a viabilidade técnica do projeto de um sistema de informação dessa natureza. Em relação aos SIGs tradicionais (CÂMARA et al. 1996), o protótipo criado dispõe de recursos para armazenar e manipular dados geográficos dos SIGs convencionais, mas recurso de análise restrito: consultas e exibição dos seus resultados sob o mapa. Com relação aos SIGs móveis (HUANG, 2019), o protótipo permite coleta, atualização, e consulta em tempo real. Sua interface facilita e torna mais conveniente as tarefas dos usuários. Além disso, o protótipo provê acesso ao banco de dados geográfico, acessa servidor de mapas, usa formato de dados intercambiável (GeoJSON), e oferece cliente móvel para consultas. A prototipação possibilitou concluir que há viabilidade técnica para implementar aplicativos com funcionalidades de SIG móvel utilizando-se as tecnologias que foram usadas e obtendo-se o conhecimento técnico gradativamente.

A pandemia de Covid-19 impossibilitou, por enquanto, avaliações em ambientes exteriores e verificar a conectividade sem fio. Um trabalho futuro será de empacotar o aplicativo de modo a prover portabilidade para diferentes sistemas operacionais de dispositivos móveis (e.g. Android e iOS). E assim integrar *hardware* e *software* (HUANG, 2019). Para tal, poderá ser usada a ferramenta *Buildozer*.

AGRADECIMENTOS

Os autores agradecem ao PIBIFSP 2020 pela concessão de bolsa discente.

REFERÊNCIAS

BUTLER, H. et al.. **The GeoJSON Format**. Internet Engineering Task Force (IETF) 2016. 28 f.

CÂMARA, G. et al. **Anatomia de Sistemas de Informação Geográfica**. Escola de Computação, SBC, 1996.

ECMA International. The JSON Data Interchange Syntax. In: **ECMA-404 The JSON Data Interchange Standard**. 2017. Disponível em: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf . Acesso em: abr 2020.

GITHUB Docs. **GitHub Documentation**. 2020. Disponível em: https://docs.github.com/pt . Acesso em: abr 2020.

HETTINGER, R. et al. **The Python Standard Library**. 2020. Disponível em: https://docs.python.org/3.7/. Acesso em: mai. 2020.

HUANG, Q. Programming of Mobile GIS Applications. **The Geographic Information Science & Technology Body of Knowledge**. 1Q Ed., 2020. John P. Wilson (ed.). DOI: 10.22224/gistbok/2020.1.2

HOLDEN, S. **The Python Wiki**. 2018. Disponível em: https://wiki.python.org/moin/FrontPage. Acesso em: 12 set. 2020.

KIVY Documentation. 2020. Disponível em: https://buildmedia.readthedocs.org/media/pdf/kivy/latest/kivy.pdf. Acesso em: 10 jun. 2020.

LARKIN, R. **Welcome to Mapview's documentation!** 2019. Disponível em: https://mapview.readthedocs.io/en/latest/. Acesso em: 08 jun. 2020.

MONGODB Manual. 2020. Disponível em: https://docs.MongoDB.com/manual . Acesso em: 08 jun. 2020.

PRESSMAN, R.; MAXIM, B. **Engenharia de Software**. 8ª Ed. McGraw Hill Brasil, 2016.

PYMONGO 3.10.1. **Documentation**. 2020. MongoDB, Inc. Disponível em: https://pymongo.readthedocs.io/en/stable/#pymongo-release-documentation. Acesso em: jun. 2020.

SUPERGEO. **Supersurv: the low cost, powerful, mobile GIS app.** 2019. Disponível em: https://www.supergeotek.com/wp-content/uploads/2019/03/SuperSurv-10.2-spec_ENG_20190320