Pinheiro and Davis Jr *Open Geospatial Data, Software and Standards* (2018) 3:9
https://doi.org/10.1186/s40965-018-0049-4

Open Geospatial Data, Software and Standards

SOFTWARE Open Access

ThemeRise: a theme-oriented framework for volunteered geographic information applications

Michele B. Pinheiro* and Clodoveu A. Davis Jr

Abstract

Motivation: Web 2.0 disseminated the possibility of engaging citizens in data collection initiatives, also known as Volunteered Geographic Information (VGI), a type of crowdsourcing. This project was conceived based on the need to quickly develop and publish geographic data collection tools and resources that could be customized to any relevant theme and engage as many contributors as possible.

Implementation: ThemeRise, a framework for generating VGI applications, was implemented with current and flexible technologies, which ensure its extensibility and evolution. The framework manages the structure and characteristics of data collection target themes individually, allowing for multi-thematic VGI initiative design. This perspective shift also allows a individualization of rewards for volunteer actions, so that the affinity of users to theme-related knowledge domains can be adequately considered and rewarded. As it currently stands, ThemeRise allows creating and publishing VGI resources in minutes, while preserving a sophisticated control over themes, contributions and user experience.

Keywords: Volunteered geographic information, Crowdsourcing, Geographic data collection

Introduction

The term Web 2.0 represents an important change in the way the Web is perceived and its products are developed, inaugurating an age in which ordinary users can freely share content. One of the consequences of Web 2.0 is the dissemination of crowdsourcing techniques, that were aggregated to GIS democratization projects [1], leading to Volunteered Geographic Information (VGI) [2] initiatives, along with other geographic crowdsourcing and crowdsensing siblings [3].

The early development of VGI applications was based on mashups of personal web pages and blogs with maps, using the Google Maps APIs. This scenario evolved to a much larger spectrum of applications, under various designations, that includes Neogeography [4], Citizen Science [5], Public Participation GIS (PPGIS) and Participatory GIS (PGIS) [6]. These terms indicate a broad interest on solving geographic data collection problems using crowdsourcing, and as a consequence initiating the need to understand and manage issues such as contributor credibility, contributed data quality [7] and accuracy [8], spatial coverage [5, 9], update frequency [10, 11] and user motivation [12, 13].

Some frameworks [14–18] speed up the task of creating VGI applications. Most are concerned with dealing with the problems mentioned before, but implement a generic structure to encompass various kinds of VGI applications. This generic structure, however, being focused around a single data collection theme at a time, precludes the exploration of specific aspects of the underlying problem. This limitation makes it hard to define more advanced controls and constraints on the contributed data. Additionally, such frameworks do not take into account volunteer preferences and level of activity, to put forth a rewards system that can be used as a motivational feature.

In this context, this paper presents ThemeRise, a theme-oriented framework for creating VGI applications. The framework manages the structure and characteristics of data collection target themes individually, allowing for

*Correspondence: mibrito@dcc.ufmg.br
Department of Computer Science, Universidade Federal de Minas Gerais, Av. Antônio Carlos, 31270-901 Belo Horizonte, Brazil

multi-thematic VGI initiative design. This perspective shift also allows a individualization of rewards for volunteer actions, so that the affinity of users to theme-related knowledge domains can be adequately considered and rewarded.

In the next section, the implementation of the framework is presented, followed by a discussion about the differences between this project and previous approaches. Next, we present a small example of its use. Finally, we present conclusions and indicate future work.

Implementation

VGI frameworks are applications that address the problem of "allowing users (generally in the role of administrators) to produce VGI applications that other users (generally in the role of volunteers) are able to contribute with geographic information and be rewarded and recognized by their work". ThemeRise encapsulates a set of structures considered relevant to produce an entire VGI application as a unit called *Theme*. In practice, a theme represents a phenomenon of interest about which there is the need to collect geographic information through volunteers. ThemeRise includes, as relevant structures, the instruments that allow collecting contributions, involving a geometric shape as spatial representation, descriptive attributes about the phenomenon of interest, and the user's reward and reputation system. The main advantage of this strategy is the ability to control each of these structures independently for each theme, a task that can be handled over to administrators in the moment of theme creation. This control adds a new level of refinement to VGI applications, allowing a diverse group of interdisciplinary administrators to combine all their themes and share volunteers in the scope of one application, using meaningful structures independently for each theme. Also, since user reputation is linked to the theme, it is possible to avoid scenarios where the rewards of different themes are considered to be the same, when in reality user contribution frequency and quality varies according to the user's relationship to each theme. For instance, contributions related to ephemeral events, such as noise pollution, are very different from contributions to persistent themes, such as map updates.

During the process of developing the framework, a set of requirements was defined and an architecture was designed considering an evaluation of tools that could be used to implement the whole application. The requirements were divided into functional and non-functional, where the functional ones describe the desired features and the non-functional ones described constraints for the implementation of features. The requirements for the development of the framework are described next, followed by an exploration of ThemeRise's architecture.

Requirements

Summarizing the problem that VGI frameworks address, the functional requirements were defined in terms of user identification and roles, theme creation, user evaluation and geographic contribution. Each of these sets of requirements will be presented next.

We consider two kinds of users: administrators, which are able to use the framework to create VGI applications (themes), and volunteers, which are able to perform contributions. Additionally, volunteers can be anonymous, which in some cases can encourage people to more securely contribute about sensitive themes. Hence, the application must implement these two roles for identified users (administrator and identified volunteer), and should allow the administrator to choose if a theme can receive anonymous contributions. Anonymity must be defined for each theme, since some themes may require different kinds of user profiles and permissions to accept contributions. Unlike identified volunteers, anonymous volunteers cannot receive rewards, and therefore do not have a reputation score.

In ThemeRise, only administrators are able to create new themes. In the process of creating a new theme, themes are identified by name and description. As said before, contribution instruments should be selected and customized as part of theme creation. Administrators must be able to define a form for volunteers to fill out when they perform a contribution on the theme. In the form, along with descriptive fields, the administrators will also define which data types will be used to store the information collected in an underlying DBMS, along with the geographic representation. The framework should allow the administrator to choose the most suitable geographic representation for the theme, which can be Point, Line or Polygon. The administrators should also set their desired visualization settings in the moment of theme creation, including the representation symbols. Additionally the framework should allow administrators to define if the theme will require complementary artifacts to enrich the descriptions, like wikis and files, or if the theme will provide instruments to support discussions about the contribution, by receiving comments on contributions.

In ThemeRise, users should be recognized by their efforts on contributing to a theme, rather than by their overall contributions to multiple themes. Therefore, the framework should allow the administrators to define how many reward points the identified volunteer should receive for performing an action within the theme.

To bring more recognition and identify the most committed volunteers, the framework should assign users to groups of expertise. Administrators must be able to define a set of rules for users to evolve across these groups using ranges of reward points for each one.

These strategies were created based on [18]. However, ThemeRise also considers the theme for evaluation. The previous work proposes a system of rewards in which the user receives points for performing positive contributions to the community, and loses points when the contributions are negative. Volunteer efforts are also recognized by a title, based on experience or derived from belonging to a group of expertise. In [18], scores are taken into account and aggregated as a single value for each user, regardless of theme.

Finally, the last group of features is related to the ability of collecting contributions by volunteers to themes defined by administrators. The framework should allow volunteers to contribute in a theme providing information according to the settings defined by the administrators in theme creation, including geometric representation and descriptive attributes, arranged in a form. Following the same idea, volunteers should also be able to create complementary artifacts like wikis, comments, and attach files according with those settings. Also, according to the idea of recognizing the volunteers by their contributions, they must receive rewards for performing these actions.

Non-functional requirements define how the features are developed, and also are taken into account when the tools are evaluated to be used. Therefore, these requirements include the extensibility and flexibility of the framework, since it should be able to expand in the future and attend to different scenarios. Non-functional requirements also refer to tool selection, and include aspects such as the existence of good documentation that ensures the maintenance of the project. For the client application should be Web-based, and it should be implemented as a single page application, which can be better optimized. RESTful services were also included as requirements, since they can encapsulate some processes like database storage, and simplify the development of the framework. Additionally, RESTful services should use an ORM library that provides DBMS-independence.

Architecture and development

The proposed framework is divided into three independent layers. This structure follows closely the RESTful architecture, as proposed by Fielding et al. [19]. This design simplifies the development effort and isolates the implementation of components, allowing further expansions, such as the introduction of new components and replacement of existing ones.

The three layers in the framework are: (1) *Persistence* layer, where user generated data are stored; (2) *Business* layer, which works as a gateway between the components of the other layers; and (3) *Applications* layer, which hosts the application code, as shown in Fig. 1. Each of these layers is described next.

Persistence layer

The Persistence layer is where data produced by all users (administrators and volunteers) are managed to be stored. The component responsible for this function in this project is a relational DBMS with spatial support, which must implement the data model required by the framework. The data model can be divided into three main structures: *Themes*, *Users* and *Reputation* (Fig. 2).

As the name suggests, the Themes module contains the classes required to describe the themes and the corresponding volunteered contributions. Therefore, the main class is the *Theme* class and contains various descriptive attributes of the contribution, along with the connection to additional artifacts, such as wiki, files and comments, as was described by the theme creation requirements in previews section (p. 3). An important point of this class is the description of the attributes which will be collected as part of the contribution, since they will be used to generate the table that will store the actual contributions for this theme. The class will also be used when the user makes a contribution, since it describes the HTML elements that need to be rendered in the contribution form.

Along with the *Theme* class, there is the *Contribution* class, which contains the common structure required by every contribution and that will be extended after a theme is created to generate a specialized version to represent a theme contribution. The common structure present in all contributions describes the information about the author of the contribution (volunteer), version, date of creation, updating and deletion. As previously mentioned, the table that materializes this class in the DBMS should be dynamically created after the new theme is created by the user and inserted in the Theme table. The *Theme* and *Contribution* classes describe, thus, a behavior that emulates the catalog system in relational DBMSs.

Another group of classes involved with a theme and its contributions, as mentioned in the "Requirements" section (p. 3), are the artifact classes, which are composed by *Wiki*, *WikiHistory*, *File*, and the *Comment* classes. The latter provides a mechanism of interaction between volunteers to discuss the content of a contribution. The classes *File* and *Comment* contain the identification of the user who was responsible for that artifact, and for the contribution and theme to which the contributed file or comment belong to. Each one of these classes contains specific attributes to ensure their purpose. *File*, for example, contains fields like file name, type and location address, and *Comment* contains a text attribute. The *Wiki* class describes the current state of a wiki from a contribution, which should be shown to the application's users. To enable the possibility of going back in wiki editing, the *WikiHistory* class was designed. It contains a reference to the editing author

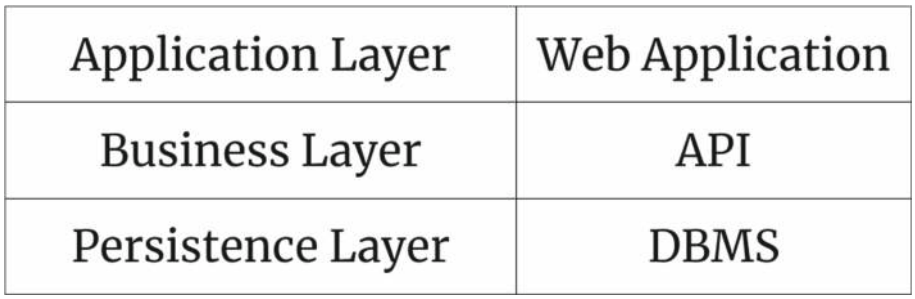

Fig. 1 ThemeRise conceptual layers

and the respective theme and contribution, along with its text.

Other then those classes, for quality measures, it is interesting to allow volunteers to rate a contribution. For this purpose, the framework includes the *Rating* class, which describes the computed rating value of a contribution. There is also a companion class, called *RatingHistory*, that describes every single user rating for a contribution.

The other module of the data model contains only the*Users* class, which represents both administrators and identified volunteers. This class manages all user information, such as name, the e-mail address used to access the application, password and dates of creation, update and deletion.

The last module is related to the reputation system, which was designed based on [18], but with a theme-oriented perspective. In this context, the reputation data module contains a class (*Action*) that describes the actions that the user can perform, and the default reward for each action. The default reward works as a suggestion for the application administrator, who can change these values for each theme in order to adjust the application to the contribution patterns that the theme can take on. The customizations made by the administrator are described by a class called *ThemeReward*. Additionally, the score earned by the user in each theme is described in the *UserScore* class, which contains references to the theme and the action performed, along with the score earned by this type of action and its numerical value.

As the user progresses and her score gets higher, the framework is designed to provide feedback in recognition of the user's involvement, by promoting her to a higher experience group. In this sense, the *Group* class was designed to describe the progress groups which can be reached by the volunteers. As in the *Action* class, this class also contains suggested threshold values for achieving a higher group. As described in the "Requirements" section (p. 3), the framework should also allow the customization of group score ranges for each theme. The *ThemePromotion* serves this purpose. It contains a reference to the group, theme, and the minimum and maximum scores that the user must achieve to progress or regress along the

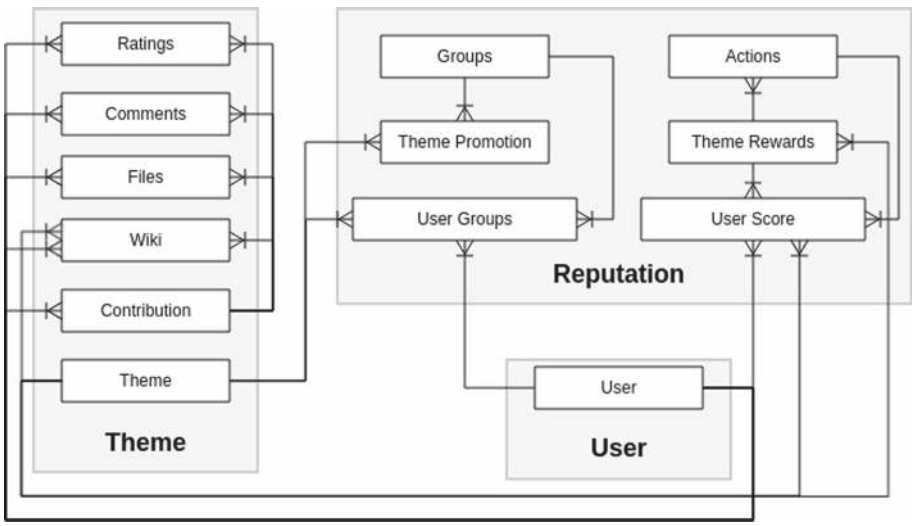

Fig. 2 ThemeRise data model

groups for each theme. There is also a class that indicates the group to which the volunteer belongs in each theme, called *UserGroup*.

Business layer

The Business layer is responsible for controlling the access to the Persistence layer and processing user-contributed information. Currently the Business layer contains only one API, but the layer's architecture allows for further expansion as the clients or the data storage get more complex. The API can be divided by protocol type, in order to deal with HTTP to fetch data from the database, as well as use web sockets to handle real-time updates in contributions, so they can be shown by the application as map updates.

The HTTP part of the API includes routes like login, registration of new users, creation and management of themes, creation and retrieval of contributions, their artifacts and ratings, and also allows editing and showing user's profiles. These routes contain three levels of access: public, authenticated and administrator, reflecting the corresponding user roles. This separation limits the access to sensitive operations, such as the creation of new themes, banishment of users, and others.

The web socket architecture uses another way of communication between the client and the service, which consists in watching and triggering events. In this design, instead of creating a short-term connection between client and service to request and receive information, with web sockets this connection is started and remains active until the client requires the end of connection, or the connection is unexpectedly interrupted by network failure. This enables the service to send new information every time it receives updates. Updates are usually sent as an event that is watched by the client. For the application, this behavior is useful, since it keeps the application map always up-to-date. Hence, the web service should implement events using web sockets to actively send current information to the client.

Considering the requirements analysis, four frameworks were considered as alternatives for the development of the API: Django and Pyramid, for Python, and Express and Hapi, for NodeJS (which was developed using Google's V8 Javascript engine). We preferred Hapi at the end of the analysis because of its good performance, flexibility and maintainability. Hapi is currently used by major companies such as Netflix and Walmart.

We also set out to select an ORM library for the framework, in order to ensure component independence from the Persistence layer. The ORM2 and SequelizeJS libraries were tested in order to verify their ability to handle data definition and queries for both spatial and conventional data types, relying on the underlying DBMS's capabilities. SequelizeJS was chosen, since it was able to handle spatial data from PostgreSQL Although SequelizeJS is only fully functional with PostgreSQL, the support for other

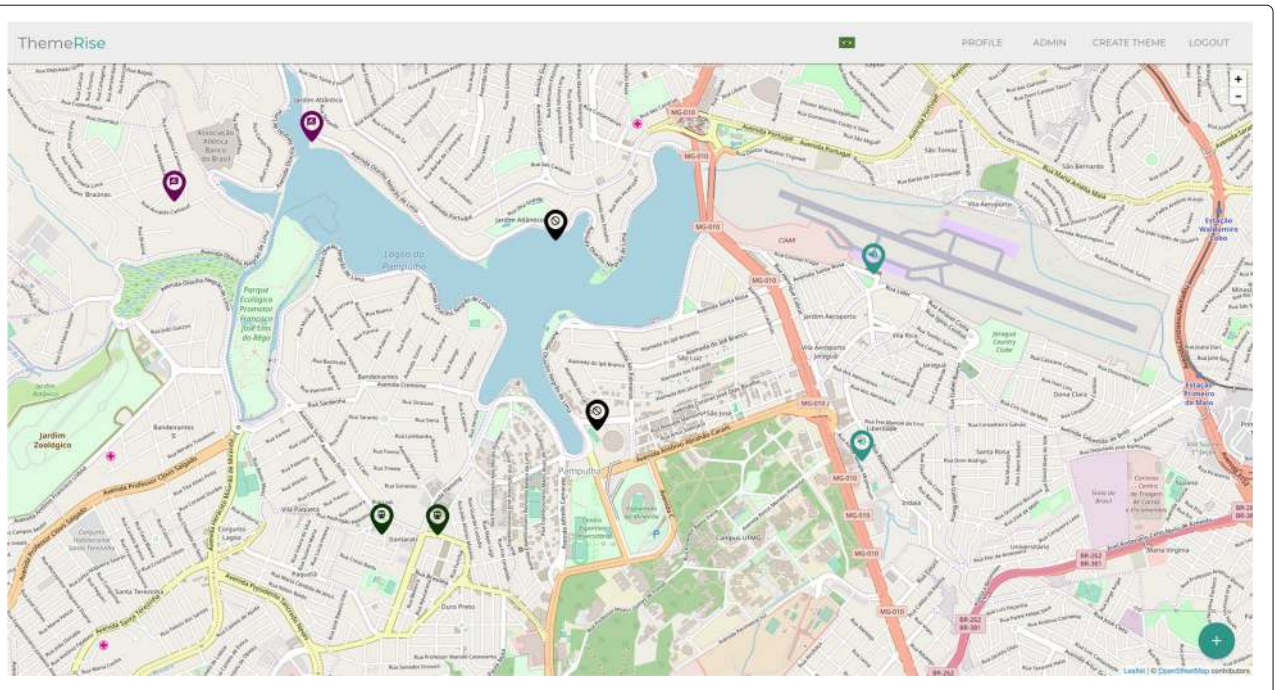

Fig. 3 ThemeRise's map for administrators

DBMSs is likely to expand, since the library is already capable of handling conventional data managed by other platforms.

The API was also implemented with a web socket element by using the Socket.io library. This library is the most popular one for NodeJS environment, and also has a client version. As mentioned before, the web socket is responsible for notifying the clients when a new contribution is made.

Application layer

The last layer is the Application layer, which contains client applications. For this project, there is only one Web client so far. This client must contain an administration section, where the application administrators can create themes and manage users, a section for registered users, where they can verify their profiles and points earned in each theme, and a section with a map that allows the volunteers to contribute. Additionally, the application must contain the mechanisms for login and registering new users.

In the Application layer, one of the requirements for a web client was to use the single page application concept in the design. There are many libraries and frameworks that can be used to create this kind of application. We considered two of them, AngularJS and ReactJS, to implement a simple blog application and verify its features. This two libraries lead to different architectures. Although ReactJS allows more control over the components and the data flow through the application, AngularJS was chosen for its simplicity in promoting the MVC architecture. Associated with AngularJS, the Angular Material library was used to speedup the user interface design process.

Since AngularJS uses the MVC pattern, the web application can be divided into views, like login, register, map, create contribution, view contribution, profile, and administration views. The first ones enable the users to access the application, and are composed by simple form views. The most important user view is the map, in which the volunteer can visualize the contributions regardless of having an account in the application. The map is created using Leaflet web mapping library, which is simpler

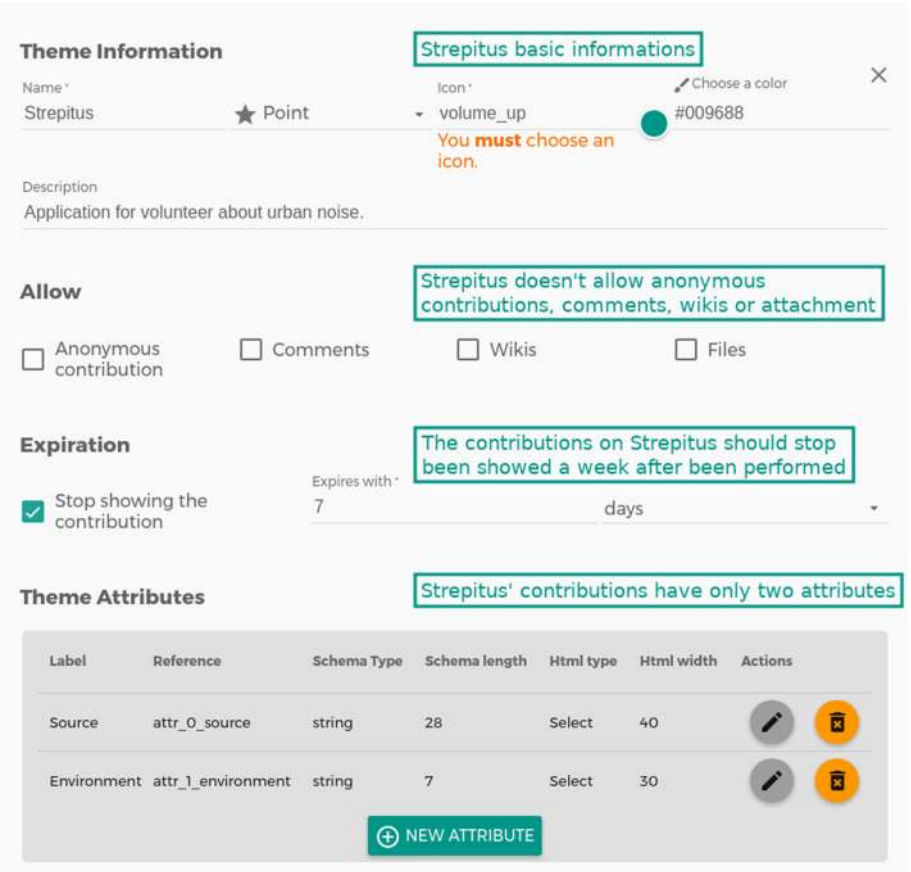

Fig. 4 View to create a new theme filled with Strepitus information

than Openlayers, but supports various plug-ins for different tasks. In the contribution part of the map, if the volunteer is not authenticated, the application will show only the themes that allow anonymous contributions. The identified volunteer can see all the themes. Additionally, a view was created for filling out the contents of the contribution, using a dynamic form that changes depending on the theme's predefined attributes. A view for the display of the contribution was also developed.

For signed-in volunteers, the application also presents a profile view, containing personal information and the current scores and groups by theme. This view is composed by a series of charts that were developed using the C3 library, which encapsulates the D3 visualization library. Besides this view, there is also a view where the identified volunteer can edit her personal information, such as email, name and password.

The administrators can access additional sections that enable them to manage themes and users. In this way, a view for creating, listing and showing the theme was developed. There are two views to manage users, one for listing and another that is similar to the profile view that the authenticated volunteers can access.

Results and discussion

In this section, a use case will be present to demonstrate the functionality of the framework. After that, we present a discussion on some specificities of this approach compared with other works already mentioned in this paper.

Use case

For the propose of demonstrating the framework, we replicated the main theme of a VGI application developed before by the authors, called Strepitus[1] [20].

This VGI application was developed from the ground up, specifically to allow volunteers to report abuses in noise generation, since this is a common environmental problem in large Brazilian cities.

Although Strepitus is an application that was implemented for both mobile and web environments, this use case focuses on replicating its Web version, with the objective of putting to use the more important features of ThemeRise, and to demonstrate and how an existing VGI application can be easily and quickly made available as a theme. The original Web application is very simple and can be thought of as a complete theme that contains only

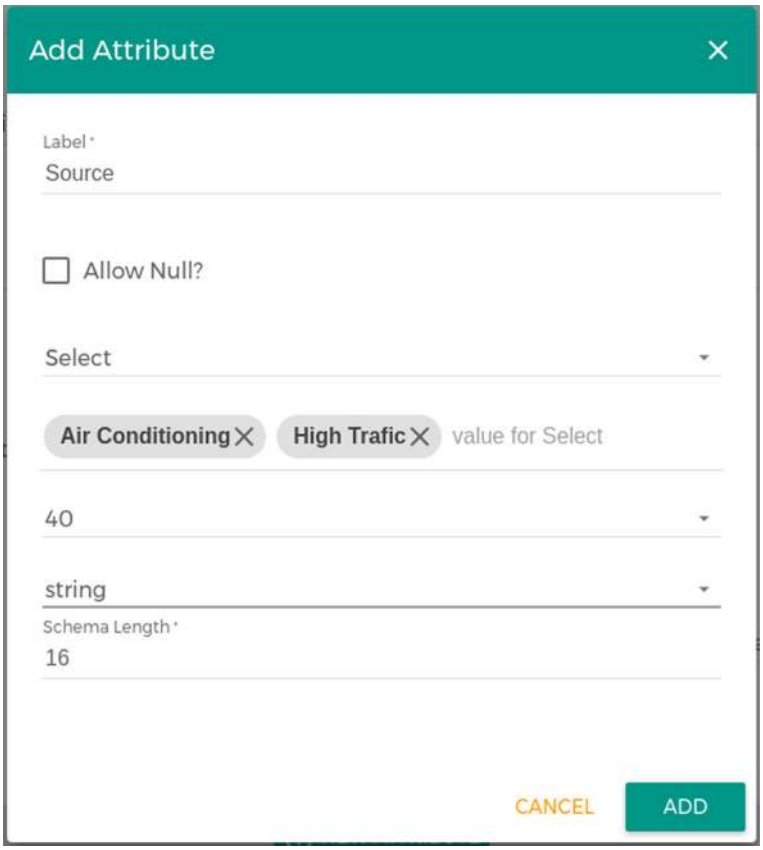

Fig. 5 Modal to include Strepitus source type attribute

two variables to be informed by the volunteers: the noise source and the environment where the noise is perceived. Another feature that can be found in the original application is the control of the display of past contributions, which is limited by an expiration time. This feature aimed to indicate the end of the noise-producing event. As an event, this feature can be configured in the corresponding theme using ThemeRise.

The first view the administrator will see contains the map and the top bar, with links to sections accessible by only this kind of user (Fig. 3). To create a new theme, the administrator must use the respective option at the menu, which will open a ThemeRise section for this purpose. This section contains a form (Fig. 4), which should be filled out with the theme's information, like its name, *Strepitus*, and its description. It also contains presentation information, such as icon, color and the geometry type, which was a point on Strepitus. Another information that is important for this theme is the expiration time, which indicates when contributions should stop being shown. In this example, we set seven days for the expiration time, as in the original application. For this example, the theme does not allow anonymous contributions, again to keep consistent with the original Strepitus application. Additionally, the administrator must inform the descriptive attributes expected for each contribution, which are an identification of the noise source (Fig. 5) and the type of environment. In the bottom of the page, the administrator can also configure the points the users will receive for each contribution and the threshold to progress among the expertise groups (Fig. 6). As previously mentioned before, these fields come pre-filled with the default values. Since the original application did not include a similar mechanism, in the example we keep these settings at their default values.

After creating the theme, volunteers are able to perform their contributions on noise pollution by clicking the green button and selecting the theme within the list shown (Fig. 7). After selecting the theme, the volunteer just needs to place a pin in the location of the contribution (Fig. 8) to

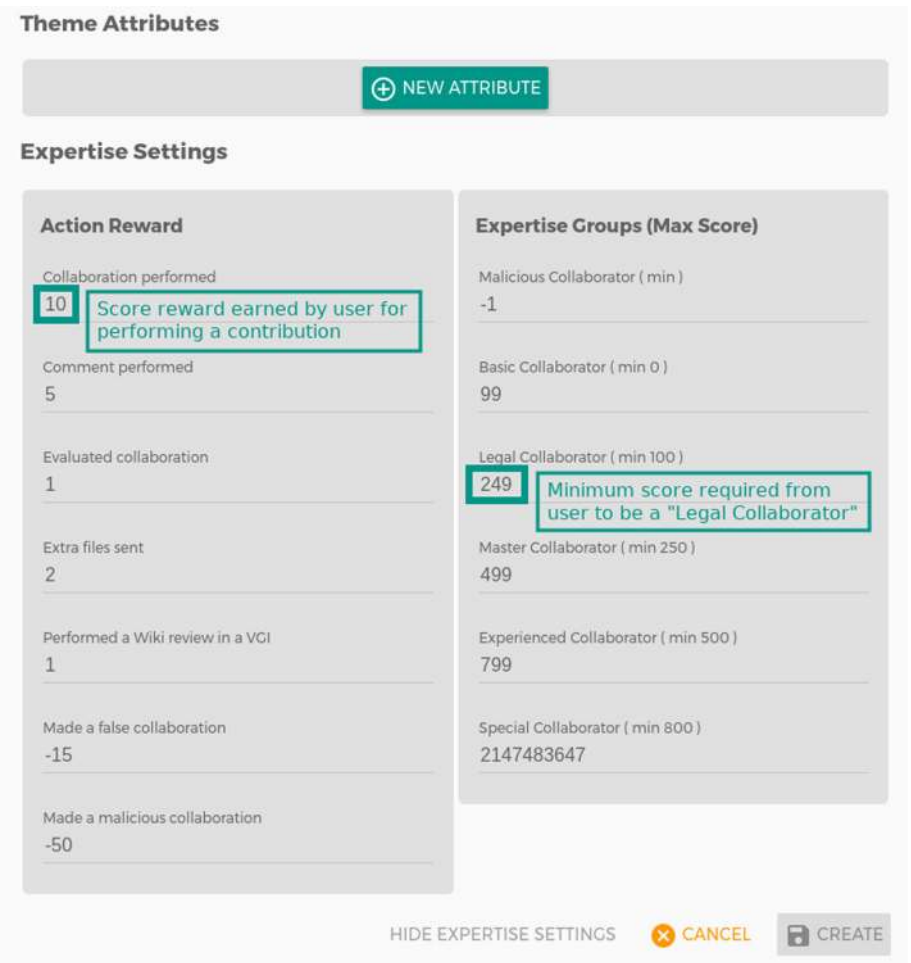

Fig. 6 Default values for user actions and groups to Strepitus

be redirected to the theme's contribution form, that can then be filled (Fig. 9). Notice, in Fig. 8, that the top bar contains only the options that are available to the contributor, as opposed to the more complete menu bar in Fig. 3.

Discussion

The theme perspective was conceptualized to provide a self-contained unit of encapsulation. In this sense, VGI applications can be composed by the replication of this unit. This shift on the perspective of VGI applications enables administrators to consider theme details, like the attributes to be collected and the points to reward users, instead of only managing standardized geographic objects and map representations. It also allows us to realize that the characteristics that differentiate Citizen Science, Neogeography, and PPGIS/PGIS VGI applications can be encoded as different themes. Each theme can be customized to reflect particular attribute structures, a different mode of volunteered evaluation, and the desired social visibility (anonymity versus personal distinction) of the contributor.

With the concept of theme in mind and its implications, there are some points in which this perspective represents a step ahead in comparison with earlier frameworks. The most significant one is the ability to structure attributes of the information collected, which contrasts with other works that focus on improving [8] or ensuring [21] information quality by controlling data types and allowed values.

The separation of themes also leads to a more accurate reputation system, which can be tuned by taking into consideration theme characteristics. The early implementations in which this work was based [18], considered one score for all "themes" collected by the volunteer. However, sometimes there are themes that demand more effort and previous knowledge from the volunteer than others. This structure could lead to discrepancies in user interest between themes, since some of them may offer a faster way to climb to higher expertise groups, that in this case reflect all themes simultaneously.

Another result obtained was the ability to host VGI projects that mix the three main types of VGI applications, Citizen Science, Neogeography and PPGIS/PGIS. The individualization of the reputation system by theme also makes this possible, since usually the time and expertise requirements in some applications are quite different from others. The application can also be configured to avoid overrating the general volunteer contributions in comparison with an expert authority contributions for the same theme.

Conclusion

This work shows a new approach for VGI application frameworks using a theme perspective. On one hand, the framework enables the administrators to structure the variables to be collected by the volunteers, which can be used to improve the quality of the data collected. On the

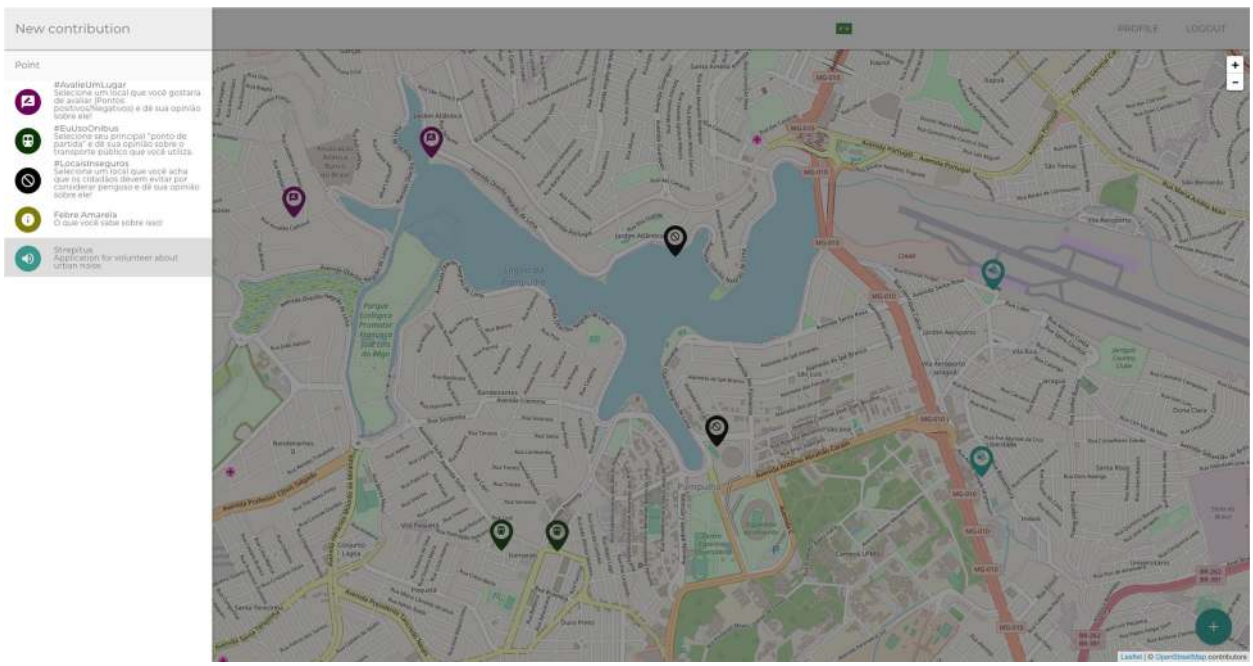

Fig. 7 List of themes to create a new contribution

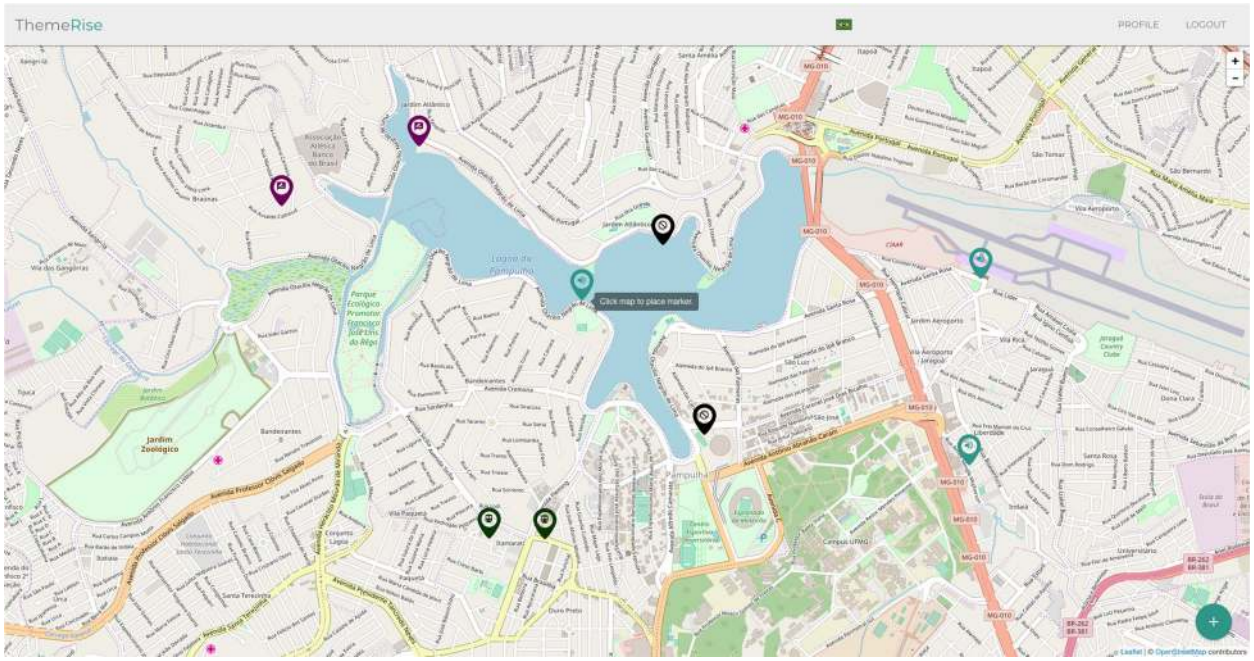

Fig. 8 Map for placing the noise pin for contribution

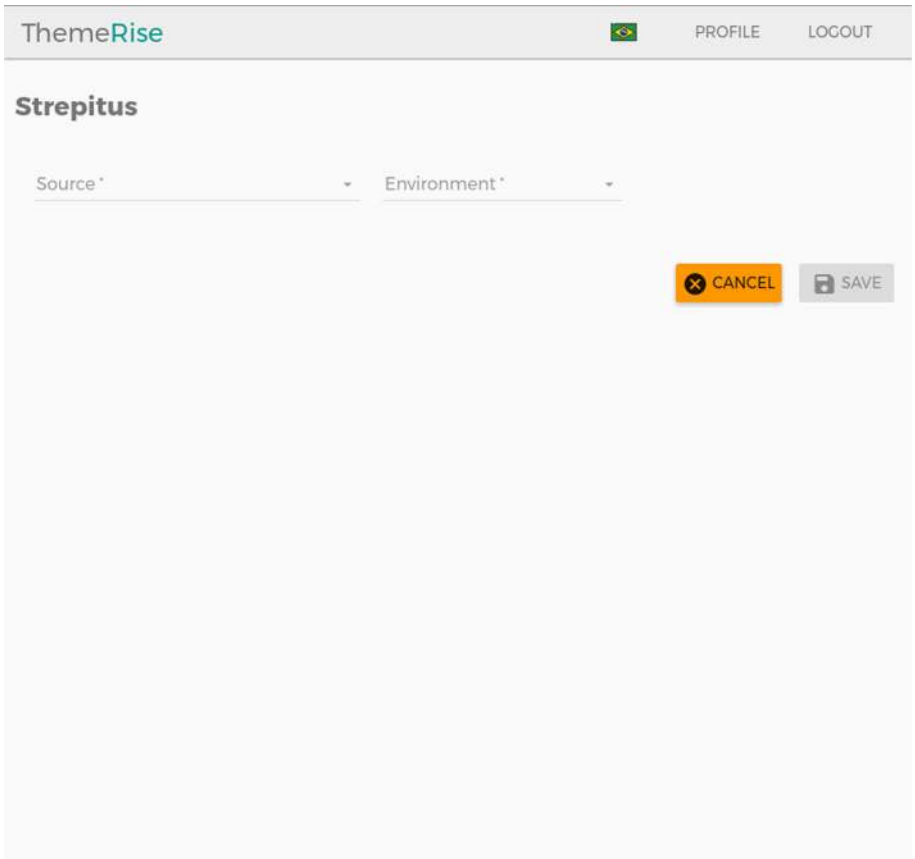

Fig. 9 Fill information about contribution

other hand, the framework also explores the idea of theme encapsulation, and applies this concept to implement a fairer reputation system, which takes into account the individual characteristics of the data collection process for each theme.

The framework was idealized to enable further improvements, which can be made by applying the theme concept to other VGI application techniques, such as gamification [22, 23] and auto-generated metadata [21]. The first one can take advantage of the reputation system and include rules in a competitive context. In this way, the various themes can help to identify gamification parameters that promote a better spatial coverage of the collection, enhance the motivation of users and obtain better results for the VGI project's target content. The auto-generated metadata, in comparison with the gamification, is simpler to structure and implement, and can lead to better quality assurance resources.

Another important concern regards the necessary improvement of quality assurance tools and techniques. Recent work [24] reviews methods to assess the quality of volunteered data based on three classification categories of VGI, defined by the authors. In this context, some of the techniques presented for map-based VGI could be coupled with the reward, reputation and gamification strategies we implemented in ThemeRise to provide more support for administrators. Other recent initiatives focus on enriching existing data sources with additional features using automatic processing of raw data collected by volunteers [25, 26]. This can be a direction for expanding the current features in ThemeRise, providing specific support for crowdsensing and crowdsourcing initiatives, directed at dataset enhancement and enrichment initiatives.

Availability and requirements

Project name: ThemeRise
Project home page: http://aqui.io/themerise
Operating system: Plataforms independent
Programming language: JavaScript
Other requirements: Node 6.2, Apache or Ngix
License: MIT

Endnote

[1] http://aqui.io/strepitus/

Abbreviations

ACL: Access control list; API: Application programming interface; DBMS: Database management system; GIS: Geographic information systems; HTML: HyperText markup language; HTTP: Hypertext transfer protocol; JSON: JavaScript object notation; PGIS: Participatory geographic information system; PPGIS: Public participation geographic information system; MVC: Model view controller; UI: User interface; VGI: Volunteered geographic information

Acknowledgements

Authors acknowledge the support of CNPq and FAPEMIG, Brazilian agencies in charge of fostering research and development.

Funding

This project was supported by individual grants to the second author from CNPq and FAPEMIG. The first author received support from CAPES, as a MSc student at UFMG.

Authors' contributions

Both authors are responsible for the overall design of the framework, and for the text of the article. MP implemented and tested the framework. Both authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 7 December 2017 Accepted: 17 April 2018
Published online: 18 June 2018

References

1. Butler D. Virtual globes: The web-wide world. 2006. https://doi.org/10.1038/439776a. http://www.nature.com/nature/journal/v439/n7078/full/439776a.html.
2. Goodchild M. Citizens as sensors: the world of volunteered geography. GeoJournal. 2007;4(69):211–21.
3. Mateveli GV, Machado NG, Moro MM, Davis Jr CA. Taxonomia e desafios de recomendação para coleta de dados geográficos por cidadãos. XXX Simpósio Bras Bancos Dados (SBBD 2015). 2015.
4. Turner A. Introduction to Neogeography. 1st ed. Sebastopol: O'Reilly Media, Inc.; 2006, p. 54.
5. Haklay M. In: Sui D, Elwood S, Goodchild M, editors. Citizen Science and Volunteered Geographic Information: Overview and Typology of Participation. Dordrecht: Springer; 2013, pp. 105–22. https://doi.org/10.1007/978-94-007-4587-2_7.
6. Sieber RE. Public participation geographic information systems: A literature review and framework. Ann Assoc Am Geogr. 2006;96(3):491–507. https://doi.org/10.1111/j.1467-8306.2006.00702.x.
7. Flanagin AJ, Metzger MJ. The credibility of volunteered geographic information. GeoJournal. 2008;72(3-4):137–48.
8. Goodchild M, Li L. Assuring the quality of volunteered geographic information. Spat Stat. 2012;1(0):110–20.
9. Sui D, Goodchild M, Elwood S. Volunteered geographic information, the exaflood, and the growing digital divide. In: Elwood S, Goodchild MF, Sui D, editors. Crowdsourcing Geographic Knowledge. Springer; 2013. p. 1–12. https://doi.org/10.1007/978-94-007-4587-2_1.
10. Neis P, Zipf A. Analyzing the contributor activity of a volunteered geographic information project — the case of openstreetmap. ISPRS Int J Geo-Inf. 2012;1(2):146–65. https://doi.org/10.3390/ijgi1020146.
11. Haklay MM, Basiouka S, Antoniou V, Ather A. How many volunteers does it take to map an area well? the validity of linus' law to volunteered geographic information. Cartogr J. 2010;47(4):315–22. https://doi.org/10.1179/000870410X12911304958827.
12. Coleman DJ, Georgiadou Y, Labonte J, et al. Volunteered geographic information: The nature and motivation of produsers. Int J Spat Data Infrastructures Res. 2009;4(1):332–58.
13. Maué P. Reputation as a tool to ensure validity of VGI. Workshop on volunteered geographic information. 2007.
14. Silva JCT, Davis Jr CA. Um framework para coleta e filtragem de dados geográficos fornecidos voltuntáriamente. X Braz Symp GeoInformatics (GeoInfo 2008). 2008:139–44. Sociedade Brasileira de Computação.
15. Okolloh O. Ushahidi, or 'testimony': Web 2.0 tools for crowdsourcing crisis information. Participatory Learn Action. 2009;59(1):65–70.
16. Sheppard SA. wq: A modular framework for collecting, storing, and utilizing experiential VGI. In: Proceedings of the 1st ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information. GEOCROWD '12. New York: ACM; 2012. p. 62–9. https://doi.org/10.1145/2442952.2442964.
17. Davis Jr CA, Vellozo HS, Pinheiro MB. A framework for web and mobile volunteered geographic information applications. XIV Braz Symp GeoInformatics (GeoInfo 2013). 2013:147–57. Sociedade Brasileira de Computação.

18. de Souza W, Lisboa-Filho J, de Sousa Câmara J, Filho J, de Paiva Oliveira A. Clickonmap: A framework to develop volunteered geographic information systems with dynamic metadata. In: Murgante B, Misra S, Rocha AC, Torre C, Rocha J, Falcão M, Taniar D, Apduhan B, Gervasi O, editors. Computational Science and Its Applications – ICCSA 2014. Lecture Notes in Computer Science. Portugal: Springer; 2014. p. 532–46. https://doi.org/10.1007/978-3-319-09129-7.
19. Fielding RT. Architectural styles and the design of network-based software architectures. PhD thesis, University of California, Irvine. 2000.
20. Vellozo SH, Pinheiro BM, Davis Jr AC. Strepitus: um aplicativo para coleta colaborativa de dados sobre ruído urbano. IV Workshop Computação Apl Gestão do Meio Ambiente Recur Naturais. 2013. Sociedade Brasileira de Computação.
21. de Souza WD, Lisboa Filho J, Vidal Filho JN, Câmara JH. DM4VGI: A template with dynamic metadata for documenting and validating the quality of volunteered geographic information. XIV Braz Symp GeoInformatics (GeoInfo 2013). 2013:1–12.
22. Antoniou V, Schlieder C. Participation patterns, VGI and gamification. Proceedings of AGILE 2014. Presented at the AGILE. 2014.
23. Martella R, Kray C, Clementini E. In: Bacao F, Santos YM, Painho M, editors. A Gamification Framework for Volunteered Geographic Information. Cham: Springer; 2015, pp. 73–89. https://doi.org/10.1007/978-3-319-16787-9_5.
24. Senaratne H, Mobasheri A, Ali AL, Capineri C, Haklay MM. A review of volunteered geographic information quality assessment methods. International Journal of Geographical Information Science. 2017;31(1):139–67. https://doi.org/10.1080/13658816.2016.1189556.
25. Mobasheri A. A rule-based spatial reasoning approach for openstreetmap data quality enrichment; case study of routing and navigation. Sensors. 2017;17(11):. https://doi.org/10.3390/s17112498.
26. Mobasheri A, Huang H, Degrossi LC, Zipf A. Enrichment of openstreetmap data completeness with sidewalk geometries using data mining techniques. Sensors. 2018;18(2):. https://doi.org/10.3390/s18020509.