A SEMI-AUTOMATIC PROCEDURE FOR A DEMOGRAPHIC ANALYSIS OF THE FOSS4G DEVELOPERS' COMMUNITY

D. Oxoli [a], H.-K. Kang [b]*, M. A. Brovelli [a]

[a] Department of Civil and Environmental Engineering, Politecnico di Milano, Milan, Italy - (daniele.oxoli, maria.brovelli)@polimi.it
[b] Geospatial Information Research Division, Korea Research Institute for Human Settlement, Sejong 30147, Korea - hkkang@krihs.re.kr

Commission IV, WG IV/4

KEY WORDS: Community, Collaboration, FOSS4G, GitHub, Software Development, Open Source

ABSTRACT:

The open and direct collaboration at the creation, improvement, and documentation of source code and software applications - enabled by the web - is recognized as a peculiarity of the Free and Open Source Software for Geospatial (FOSS4G) projects representing, at the same time, one of their main strengths. With this in mind, it turns out to be interesting to perform an extensive monitoring of both the evolution and the geographical arrangement of the developers' communities in order to investigate their actual extension, evolution and degree of activity. In this work, a semi-automatic procedure to perform this particular analysis is described. The procedure is mainly based on the use of the GitHub Search Application Programming Interface by means of JavaScript custom modules to perform a census of the users registered with a collaborator role to the repositories of the most popular FOSS4G projects, hosted on the GitHub platform. The collected data is processed and analysed using Python and QGIS. The results - presented through tables, charts, and thematic maps - allow describing both dimensions as well as the geographical heterogeneity of the contributing community of each individual project, while enabling to identify the most active countries - in terms of the number of contributors - in the development of the most popular FOSS4G. The limits of the analysis, including technical constraints and considerations on the significance of the developers' census, are finally highlighted and discussed.

1. INTRODUCTION

The concepts of community and participation - together with the dogmas of the free dissemination and use of content - represent the pillars of the open source movement. These principles as well apply in the context of the Free and Open Source Software for Geospatial (FOSS4G) development (Brovelli et al., 2017). The open source communities are typically founded by individuals or groups which operates autonomously while recruiting and networking with other communities or community members to contribute organically to the common goal of developing and maintaining software projects of interest (West & O'mahony, 2008).

Community software development is generally performed through the use of distributed Version Control Systems - such as the Git (https://git-scm.com) - which enable the management of source code creation, revision, and deployment among large groups of contributors (Alwis & Sillito, 2009). The Git framework is implemented in different web platform allowing contributors to host and manage their development work on cloud-based systems. Nowadays, one of the largest and most popular Git-based web-hosting platforms is GitHub (https://github.com) (Kalliamvakou et al., 2014).

This distributed and participatory development approach unfolds a general interest in discovering both geographical extension and degree of activity of the developers' communities (Thung et al., 2013). Therefore, the primary goal of the presented work is to outline the contribution of individuals and national communities to the development and/or maintenance of the most popular FOSS4G projects. Being the ecosystem of FOSS4G extremely broad and heterogeneous, the most representative projects have been selected considering the ones promoted by the Open Source Geospatial Foundation (OSGeo, https://www.osgeo.org), a not-for-profit organization which mission is to foster the adoption of open geospatial technology worldwide by supporting community-driven software development. This is achieved by exploiting the GitHub platform capabilities, where indeed a number of the OSGeo projects are hosted by means of dedicated source code repositories (Löwe et al., 2017).

The paper is structured as follows: Section 2 presents details and limitations of the data collection strategy adopted. Results are presented in Section 3. In Section 4, conclusions and future directions for the work are outlined.

2. DATA COLLECTION STRATEGY

Besides the hosting facilities, GitHub provides a number of Application Programming Interfaces (APIs) which allow users to programmatically interact with the platform services. Of primary interest for this work are the capabilities enabled by the GitHub Search API (https://developer.github.com/v3/search). The Search API provides developers with functionalities for inquiring items within GitHub such as users' profiles and/or project repository features.

*Corresponding author

A dedicated software application to perform the presented procedure - based on custom JavaScript modules and Python scripts - has been developed by the authors and made available on GitHub (https://github.com/danioxoli/FOSS4G_contributors). A short guide for running the data collection and analysis can be found in the above repository, by easing the reuse of the application.

2.1 Data collection and processing

The Search API has been exploited to automatically retrieve from the GitHub platform first, the number of users which have contributed to each of the OSGeo selected projects (see Figure 1) and last the location of each identified user by querying its personal GitHub profile. The location information is among the optional parameters which users are free to specify at the creation of their personal GitHub accounts. Therefore, location information is not always available for the whole identified users.

The designed procedure for discovering contributors of each project of interest requires as input a list of GitHub project repositories in CSV format including their URLs. A dedicated JavaScript module is run to perform the collection of contributors' GitHub usernames for each repository included in the input CSV list by compiling and sending specific Search API requests. The module combines API responses into a single JSON dictionary containing as keys the repository names and as values the list of detected contributors' GitHub usernames. The output of this first step is passed in input to a second JavaScript module which queries one-by-one the GitHub profiles connected to each contributor's username - again by means of Search API requests - to collect the location information, when available. This returns a JSON dictionary of dictionaries in which contributor-location information is nested in the values of the repository-contributors dictionary.

Collected data is pre-processed by means of Python in order to extract information such as a) the number of contributors by project, b) the number of contributors by country, and c) the share of contributors to each project by country, thus enabling an overview of the FOSS4G contributors' community through simple maps and graphs. The JSON dictionary obtained from the previous step is parsed and fit into a Python Pandas Data Frame, which is a tabular data format providing powerful data manipulation, analyses, and visualization capabilities. The Data Frame is designed to contain one row for each detected contributor while having as columns its attributes such as the contributing project and the location, when available.

Contributors' locations are available as toponyms. Due to the nature of this information, which is a not mandatory requirement for the user registration on GitHub, toponyms are often very general depicting only the country of origin or the city at most. Therefore, the minimum scale selected for contributors' census analysis is conveniently set to the country boundaries. To pass from toponyms to geographical coordinates, which are needed to aggregate contributors by country using any GIS software, geocoding is used. The geocoding is applied to the entire Data Frame by using as input the location column and storing the geocoded coordinates, as WGS84 Lat and Long, in two new columns. The Python GeoPy library is used to perform this operation. This library allows using within Python different geocoding APIs form many providers. For this work, Google Geocode API (https://developers.google.com/maps/documentation/geocoding) is employed.

The Data Frame is finally stored in a CSV report including in each record the contributor username, the connected contributing project, and the geographic coordinates for the geocoded contributors. Besides a full export containing all the detected contributors with location coordinates when available, other exports are created to account for only geolocated contributors as well as unique contributors. This latter because the same contributor might contribute to more than one project by biasing the count of contributors per country. A summary of the collected data is included in Table 1.

Data	**Count**
Queried GitHub project repositories	29
Entries for the contributors' collection *(including multiple counts for contributors to more than one project)*	1546
Geocoded entries for the contributors' collection *(including multiple counts for contributors to more than one project)*	864
Geocoded unique contributors	675

Table 1. Global counts for data considered in the analysis (collection performed on January, 16th 2018)

2.2 Data limitations and representativeness

Main limitations of the data collection procedure are due to a) the incomplete information for the contributors' locations, which nevertheless depends on the personal account settings of each contributor, and b) the request-rate limit of the GitHub Search API, which is set at 5000 requests per-hour for authenticated users. For the presented work, the API request-rate limit was not achieved. Potentially, the designed procedure can be adapted for querying any list of GitHub repositories. In case of application to a longer list of repositories as well as to repositories with larger contributors' group, the API request-rate limit might prevent the procedure to succeed.

Another important consideration is related to the completeness of collected data. Regarding OSGeo projects, not all the source code of every software is hosted on GitHub. The proposed procedure is based on APIs which are not equally available for other popular Git web platforms. This makes the use of GitHub the only suitable solution to programmatically perform contributors' data collection such as the one here presented.

Moreover, for some of the considered projects there exist multiple GitHub repository on which development work is carried out. This work is limited to the analysis of the official repositories listed on the OSGeo website. With this in mind, the collected data has to be intended like a sample of the whole contributors' community which representativeness might vary from project to project.

3. RESULTS OVERVIEW

The collected data, after the geocoding and post-processing operation described above, is analysed by taking advantage of Python and QGIS to produce synthetic and exhaustive data presentations by means of graphs and thematic maps. Data is aggregated by project and by country to extract indicators such as global counts and shares of contributors within both countries and projects. The number of contributors is also compared with country population. Results are presented in the following.

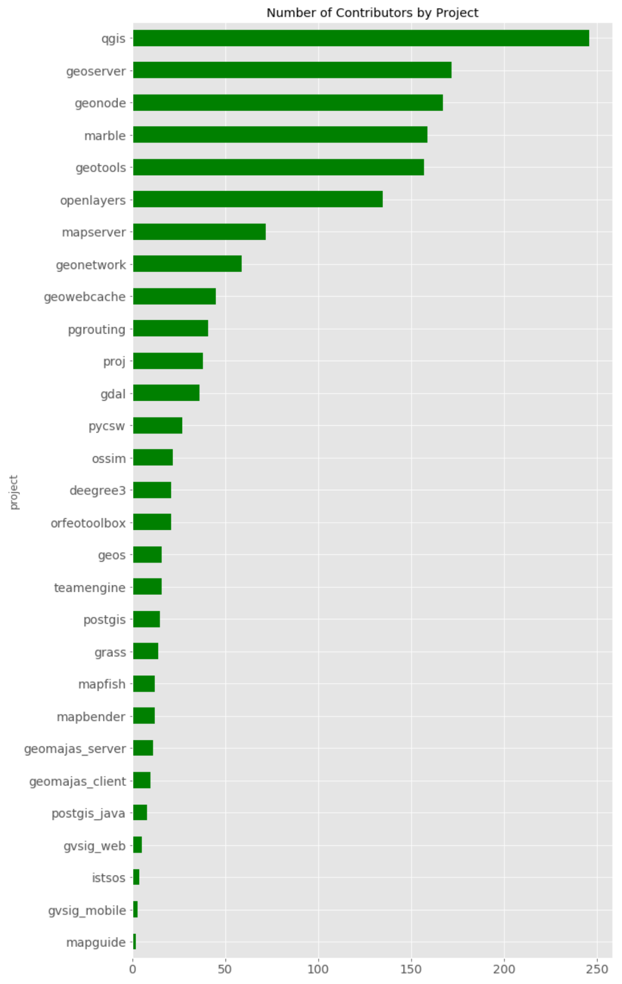

Figure 1. Count of total contributors by project

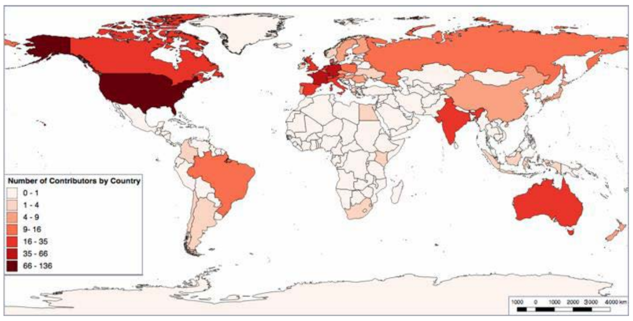

Figure 2. Map of geocoded unique contributors by country (world country map by GADM: http://gadm.org)

According to the analysis results, in descending order, the top-five projects by number of contributors are: QGIS, GeoServer, GeoNode, Marble and Geotools (see Figure 1). The top-five countries by number of contributors are: USA, Germany, France, Italy and the United Kingdom (see figure 2 and 3).

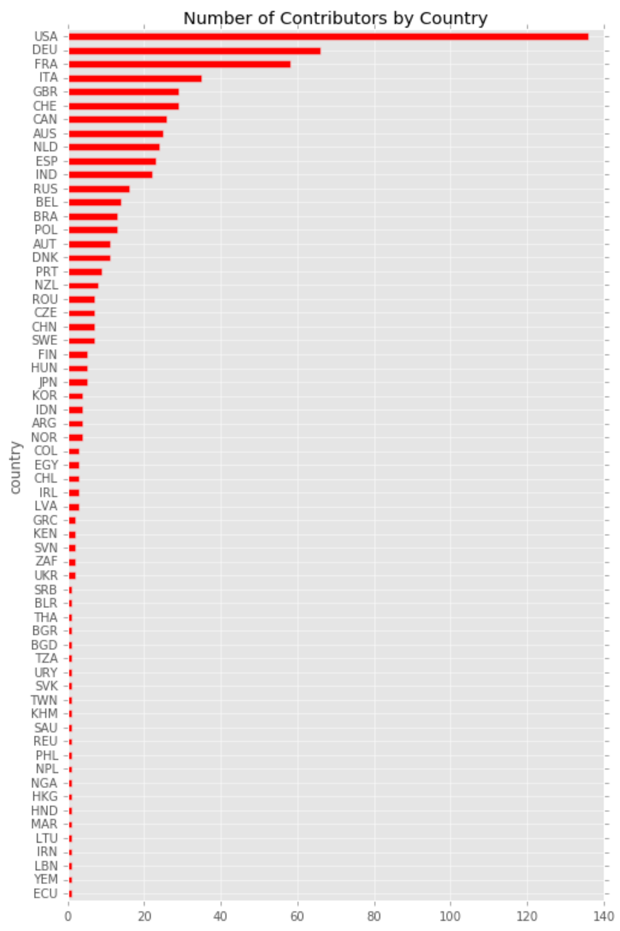

Figure 3. Count of geocoded unique contributors by country (country code: ISO alpha-3)

By considering the contributors share out of every 1000 inhabitants, the top-five countries are instead: Switzerland, Denmark, New Zealand, Latvia and the Netherlands (see figures 4 and 5). Finally, the distribution of county contributors by project is summarized in Figure 6.

As an example, the Republic of Korea is placed at the 27th position by number of contributors and at 40th position by contributors share out of every 1000 inhabitants. The total count of detected Korea's contributors is four, which are active in four different projects, namely: QGIS, GeoNode, Geotools and OpenLayers.

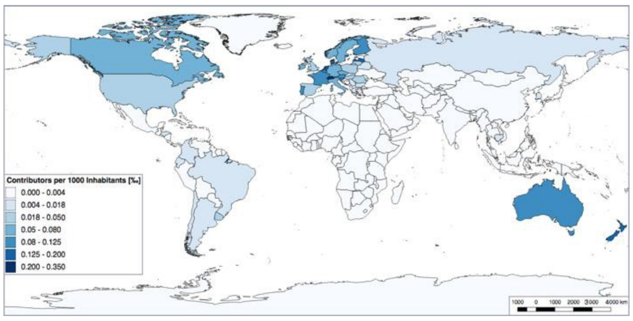

Figure 4. Map of geocoded unique contributors share out of every 1000 country inhabitants [‰] (world country map by GADM: http://gadm.org)

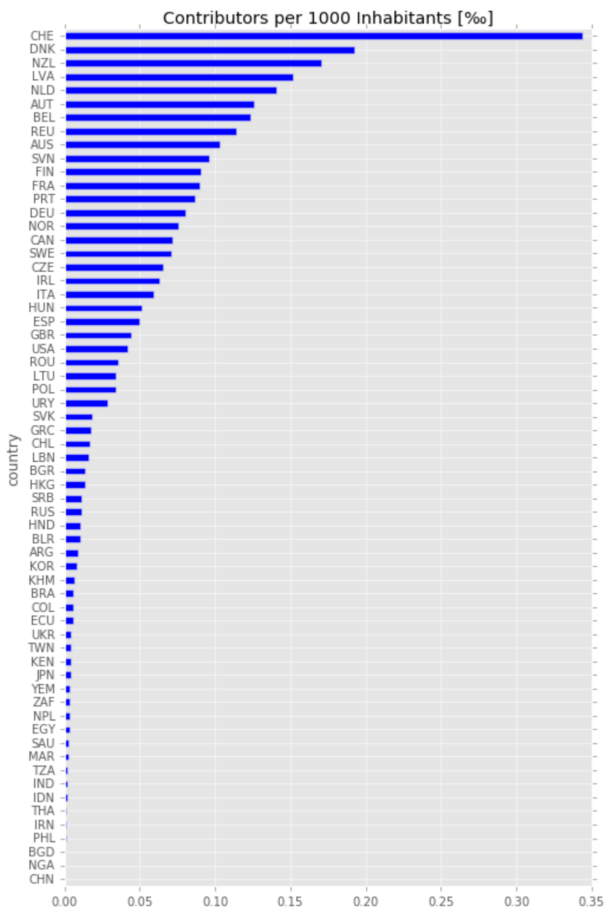

Figure 5. Share of geocoded unique contributors out of every 1000 country inhabitants [‰] (country code: ISO alpha-3)

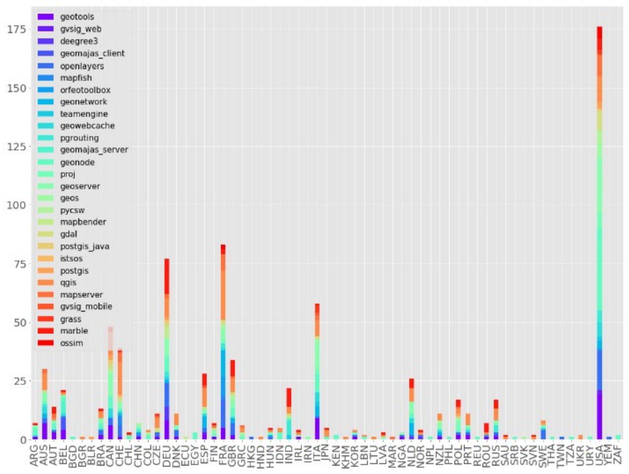

Figure 6. Count of country contributors by project (country code: ISO alpha-3). The Mapguide project does not appear in the legend because none of its contributors was geocoded

4. CONCLUSIONS AND FUTURE WORK

A semi-automatic procedure for collecting census data of the OSGeo developers' community was outlined. Results here presented help in understanding both the dimension and the geographical distribution of the contributors' community providing snapshots of both the projects ranking in terms of participation and the geographical distribution of the community.

Collected data has to be intended as a sample of the whole OSGeo developers' community because of the limitation of the presented collection strategy discussed in Section 2.2. Representativeness might vary from project to project depending on their share of source code actually hosted on the GitHub platform. The choice of the input repository list is therefore critical in order to collect a significant amount of representative data. The source code of the software application developed for this work is available with an open license on GitHub, thus enabling future improvements as well as applications of the presented analysis.

Future work will focus on extending and repeating the analysis by including FOSS4G projects which were not considered in this initial case study. The feasibility of including in the census also project repositories which are hosted on platforms different from GitHub will be also investigated.

ACKNOWLEDGEMENTS

This research was founded by the Geospatial Information Research Division of the Korea Research Institute for Human Settlement (http://eng.krihs.re.kr).

REFERENCES

Alwis, B. D., & Sillito, J. (2009). Why are software projects moving from centralized to decentralized version control systems? In: *2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, pp. 36-39.

Brovelli, M. A., Minghini, M., Moreno-Sanchez, R., Oliveira, R. (2017). Free and open source software for geospatial applications (FOSS4G) to support Future Earth. *International Journal of Digital Earth*, 10(4), pp. 386-404.

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., Damian, D. (2014). The promises and perils of mining github. In: *Proceedings of the 11th working conference on mining software repositories*, pp. 92-101.

Löwe, P., Neteler, M., Goebel, J., Tullney, M. (2017). Towards OSGeo Best Practices for Scientific Software Citation: Integration Options for Persistent Identifiers fn OSGeo Project Repositories. In: *Free and Open Source Software for Geospatial (FOSS4G) Conference Proceedings*, 17(1), pp. 135-145.

West, J., & O'mahony, S. (2008). The role of participation architecture in growing sponsored open source communities. *Industry and innovation*, 15(2), pp. 145-168.

Thung, F., Bissyande, T. F., Lo, D., Jiang, L. (2013). Network structure of social coding in GitHub. In: *17th IEEE european Conference on Software Maintenance and Reengineering (CSMR 2013)*, pp. 323-326.