Developer Reputation Estimator (DRE)

Sadika Amreen*, Andrey Karnauch† and Audris Mockus‡
Department of Electrical Engineering and Computer Science
The University of Tennessee, Knoxville, TN, USA
Email: *samreen@vols.utk.edu, †akarnauc@vols.utk.edu, ‡audris@utk.edu

***Abstract*—Evidence shows that developer reputation is extremely important when accepting pull requests or resolving reported issues. It is particularly salient in Free/Libre Open Source Software since the developers are distributed around the world, do not work for the same organization and, in most cases, never meet face to face. The existing solutions to expose developer reputation tend to be forge specific (GitHub), focus on activity instead of impact, do not leverage social or technical networks, and do not correct often misspelled developer identities. We aim to remedy this by amalgamating data from all public Git repositories, measuring the impact of developer work, expose developer's collaborators, and correct notoriously problematic developer identity data. We leverage World of Code (WoC), a collection of an almost complete (and continuously updated) set of Git repositories by first allowing developers to select which of the 34 million(M) Git commit author IDs belong to them and then generating their profiles by treating the selected collection of IDs as that single developer. As a side-effect, these selections serve as a training set for a supervised learning algorithm that merges multiple identity strings belonging to a single individual. As we evaluate the tool and the proposed impact measure, we expect to build on these findings to develop reputation badges that could be associated with pull requests and commits so developers could easier trust and prioritize them.**

Link to demo video – https://youtu.be/KyfaXtv7hA8

Link to source code – https://github.com/ssc-oscar/DRE

***Index Terms*—Developer Reputation, Software Ecosystem, Identity Disambiguation**

Supported by NSF awards 1633437 and 1901102.

I. Introduction

Software development is a highly collaborative activity. The version control system Git and "social coding" platform GitHub, for example, support collaboration through code contribution, code sharing and knowledge exchange, and Free/Libre Open Source Software (FLOSS) software development is no longer bound within small groups or communities of developers. FLOSS developers can work together from any physical location, during any time of the day, have any educational background, and contribute to multiple projects with varying degrees (e.g. adding features, writing patches, enhancing documentation). This talent pool of software developers is vast and highly varied in the types and amounts of experience. That makes it difficult to find collaborators with sufficient and specific expertise.

To aid in this process, a reputation measurement tool is needed for the size and diversity of FLOSS [7]. GitHub, for example, provides developer profiles that show their activity over all projects on GitHub. Expertise Browser [3] was used to support globally distributed development by both showing the proportion of developer's or organization's commits over the entire code base. However, serious gaps remain in the existing reputation estimation tools. First, none of them collect data from the entirety of FLOSS projects, and even though GitHub contains the bulk of such projects, many important projects are not hosted or mirrored there. Making measurements based on a comprehensive collection from diverse sources is, therefore, our first aim. Second, none of the existing tools show the impact of developers' work. Google Scholar, for example, does not only show the papers an author wrote but also provides the number of citations for each paper (an indication that the work was used or a measure of its impact). Providing a measure of impact developers have on other FLOSS projects and developers is, therefore, our second aim. Third, existing tools have limited social network capabilities that do not inform how important or impactful a developer's collaborators are. Thus, building a more comprehensive display of social network including important collaborators and shortest path to celebrities is our third aim. Longer term, we expect this or similar tools to evolve into trusted reputation measurement tools that could be deployed as badges to help, for example, prioritize pull requests. Finally, we would like to address a serious research problem in software development that involves data correction, especially developer identity data that is of rather poor quality in Git commits [2]. One of the biggest challenges is the lack of large labeled datasets that could be used to train supervised machine learning models to do error correction. Since survey response rates are low, constructing such data sets is a challenge. This tool, if it becomes popular, would address that problem and help research on FLOSS in general.

Our approach is to use the World of Code (WoC) infrastructure [1] to obtain over 34M developer identities used in over 1.6 billion(B) commits gathered from over 73M non-forked repositories. Developers start by searching through these 34M identities for the ones they used in their commits. Once they select the IDs that belong to them, the tool runs a job in the background to identify all commits made using these IDs and, using the WoC infrastructure, obtains all files modified by these commits, repositories where these commits occur, and blobs created by these commits. Further calculations are then done to construct the social network and measure the impact as described below. This profile consolidates their work (in terms of development languages, commits, projects and files) across platforms and finds their collaborators with their corresponding projects, all displayed in an interactive

978-1-7281-2508-4/19/$31.00 ©2019 IEEE
DOI 10.1109/ASE.2019.00107

dashboard.

In the rest of this paper, we describe the intended users of our demonstrated tool, how to use it, the software engineering challenges we faced in construction of this tool and finally, our planned evaluation studies.

II. Intended Users

DRE is designed and built keeping a number of users and challenges in mind. For example, a service that consolidates developer activity, much like scholarly activity showcased on Google Scholar Profile or work history as on LinkedIn, is not available for software engineers. Measures such as the researchers total citation count, the h-index etc. showcase the level of expertise of an academic scholar in a holistic sense, and a more granular measure of citation count for a given scientific publication showcases the impact of the researcher in a particular domain. Similarly, a number of measures such as a developer's commit count, the number of projects the developer has worked on, the languages the developer uses for coding and the number of collaborators can indicate a developer's expertise in a domain. Furthermore, a developer's impact can be measured by calculating the extent of reuse of the code created by the developer. Here, we outline the potential users of the implemented DRE and the benefits it can bring them.

Individual Developers: DRE aggregates a developer's contribution from multiple forges (and from multiple author IDs found in Git commits) and creates a consolidated measure of impact and enables showcasing their areas of expertise in the form of a personalized profile. The profiles of users can be helpful in assessing their centrality, reach and contribution to FLOSS. From the user's standpoint, this tool can provide evidence to support their stature of professional expertise as well. Summarized data can be used to create reputation badges which can act as a signal for the quality of the work they are doing and, therefore, increase the confidence that their issues and/or pull requests are of high quality. DRE enables developers to quench their curiosity regarding their contribution, impact, social centrality etc.

Researchers: The software engineering community is going through an evolution of practices with contributors adopting social platforms for development and knowledge exchange. As a result, researchers in commerce and academia are trying to understand the participation and contributions of developers in this collaborative environment. Mining software repositories help researchers answer a number of questions, such as "Who are the most influential developers?", "What is a developer's contribution across platforms?" [6], "What are the roles of the various contributors?" [4]. Unearthing answers to these questions help researchers understand the OSS development process better in terms of realizing who's driving the evolution process and who are the central people to a software project. To find answers to these, a complete set of data containing software development activities across all platforms is required as realized in, for example [1]. For any research to be effective, the data also needs to be free of errors, especially errors related to misrepresentations of the the developer identity. DRE allows researchers to upload and correct a list of developer IDs through advanced methods of disambiguation carried out in the backend and validated by the developers themselves.

Recruiters: Seeking expertise required for a particular software development task is a challenging process. While job seekers may claim proficiency in various languages and domains, verification of the claim requires large amounts of resources such as time (for lengthy interviews) and expert judges of performance. An online profile of developers can act as a bridge between job seekers and potential employers by corroborating claims of expertise. Data from OSS repositories have been useful for discovering technical expertise in the past [5]. For example, a developer's contribution to a project in terms of commits and number of commits made in various languages can corroborate claims made in a resume by a job seeker. DRE reserves the capacity to introduce badging metrics in the future that can provide further confidence to recruiters.

III. Software Engineering Challenges

Many tasks in software engineering evolve around modeling and supporting developer activity. Software developers can participate in multiple projects at a given time in various ways such as sharing code and collaborating and exchanging knowledge through question/answers or documentation. Measuring such activity, especially the more complicated measures such as impact, is highly complicated. We start from author IDs as used in Git commits, determine all projects (and all other developers who worked on these projects) the commits belong to, and all files and blobs created by these commits. The blobs authored by a developer are scrutinized for impact as described below. We also calculate the full bi-partite graph (with 34M developers and 73M projects) of developer to project (repository) links which we use to identify collaborators and to calculate the shortest paths.

Data collection: It is not possible to see an individual's global impact if the activities of the individual are scattered across multiple platforms. Software development activity data poses this problem as it is dispersed across multiple forges, version control tools, issue trackers etc. and is extremely large in volume. In order to showcase all contributions of a particular individual, a full collection of data that is merged from all platforms is required. Collecting and hosting such huge volumes of data is extremely time and space consuming. We utilize the WoC infrastructure to build our tool. As described above, the WoC mines this data from various version control systems, that serve as the backend for this online tool.

Data quality: The extremely large volumes of data collected from various forges and version control tools are of low quality. Particularly, this data is ridden with identity errors in the form of synonyms (a single developer with multiple IDs) and homonyms (a single ID that may be used by multiple developers and carry no information pertaining to the developer). This makes consolidation efforts extremely challenging as each developer is represented through various credentials across and within platforms. In an effort to correct

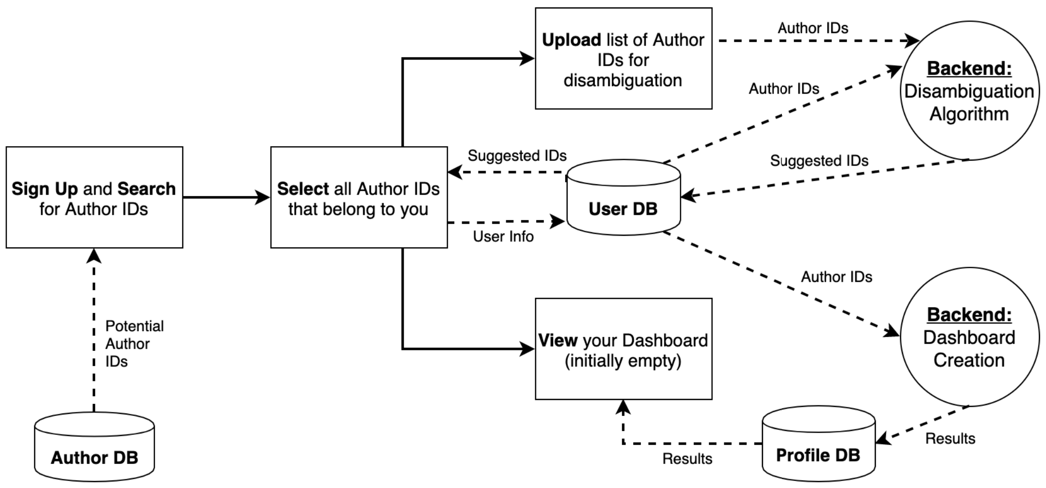
Upload list of Author IDs for disambiguation
Author IDs
Author IDs
Backend: Disambiguation Algorithm
Suggested IDs
Suggested IDs
Sign Up and Search for Author IDs
Select all Author IDs that belong to you
User DB
User Info
Author IDs
Potential Author IDs
View your Dashboard (initially empty)
Backend: Dashboard Creation
Author DB
Results
Profile DB
Results

Fig. 1. Layout of the Developer Reputation Estimator (DRE): Front and back end

these errors, DRE leverages the disambiguation framework ALFAA [2], that finds matches between developer names, email addresses and other activity traces found in the commit using supervised machine learning techniques.

Data size: To locate the initial set of author IDs, an author database of over 34M IDs is queried against several search parameters provided by the user. This process is done in real time and users expect results within seconds. Another time-sensitive operation comes soon after author selection, where a user waits for the backend to perform aggregations and network calculations on graphs of hundreds of millions of nodes and edges. To address these concerns, the databases are sharded across multiple machines for higher throughput and faster queries. For the expensive backend calculations, an initial set of blacklisted entities (e.g. a Git project that is trying to reach 1M commits artificially) that mislead graph traversals and kill computation time has been created.

IV. DRE: Usage and Capabilities

To use the tool, new users create an account by providing an email address and a password for the account. Once an account is created the users are requested for additional information such as the user's first and last names, any additional email addresses and user names (i.e. email handles, GitHub user names etc.). The search returns a list of IDs that are potential matches for the user based on exact string comparisons of the search parameters provided by the user. The user is then requested to select all IDs that belong to them.

After selections are made, the data is stored in one of the three MongoDB collections: `Author`, `User`, and `Profile`. The `Author` schema is built from information on WoC and contains the first and last names, email, user name and an author ID (name<email>). The `User` collection is populated during account creation and the schema includes the user's email, password hash, search parameters, selected IDs and omitted IDs (all IDs from the search not selected by the user), and last updated date (determines whether or not a new set of calculations is needed for this user). The `User` database feeds the selected and omitted IDs directly to the backend for its main operations. The selected and omitted IDs are used by the backend in two ways: (1) A dashboard is populated with a number of measures of the user's activity, a result of the amalgamation of all activities of all selected IDs, as we discuss below and (2) An input to the disambiguation algorithm that compares all the selected IDs with all other IDs in the dataset and predicts an outcome of match or non-match using a supervised learning technique described in [2]. While account creation is the main trigger of backend calculations, any update to the user's selected and omitted IDs (e.g. as a result of the disambiguation algorithm) is also recorded, indicating a need for re-calculation. Once the backend operations finish, the `Profile` database is populated and/or updated with the measures listed below. The tool, a layout of which is depicted in Figure 1, will be notifying the users via email that their interactive dashboard is ready for viewing which displays the following measures and capabilities.

Number of Commits, Projects, Collaborators and Files: The number of commits help show the overall lifetime activity of a developer. A higher project count may be a reflection of the expertise, passion and drive of a developer. The collaborator count can reveal a developer's willingness to work with others. The number of files modified can show the developer's impact on the OSS ecosystem.

List of Projects: An interactive list of projects showing project names, number of commits made by the user in the particular project and the total number of commits made in the project. This can help assess contribution of the user for each project.

Distribution of Coding Languages: A pie chart showing the coding languages used by the user and the number of files associated with each language in the user's commits. This shows the user's language proficiency and preference.

List of Collaborators: A consolidated list of collaborators, across all projects and platforms. Each collaborator's commit count (a measure of contribution) for a given project and the total number of commits in that project is provided as well to demonstrate the productivity of the collaborators and the size of the projects they work on.

Torvalds Index: Like Erdös number[1], the Torvalds index is the shortest path, with collaborators as nodes and projects as edges (seen on mouse-over), to Linus Torvalds, the creator and developer of the Linux kernel. Apart from being cute, this index allows the user to measure his/her position in the social network graph defined by people they collaborate with. It can provide further insight about a user's collaborators, revealing projects or persons that may be of interest to the user.

List of Blobs: Google Scholar Profiles show the impact of an academic through citation count. Similarly, we suggest to measure an aspect of the impact of a developer through the number of reuses of their code. For each blob created by a developer's commit, we first determine an author for that blob. To obtain this author, all commits creating the blob are gathered and the first commit (in calendar time) is identified. The author of that commit is considered to be author of the blob. In the impact measure, only blobs where the developer is the author (as defined above) are considered. For each such blob we count the number of commits done by other developers. Such commits indicate that these other developers have copied the original author's code for use in their projects: similar to academic citations indicating the use of others' results. The user is shown a list of blobs which they have created along with a number of commits done by others creating the same blob, and authors of these commits. The duplication is a measure of the impact of the user, indicating how many times the blob has been copied. The authors of these blobs allows the user to see their impact in the social network graph by observing which of their collaborators (if any) used their code/blob.

Upload list of IDs: Researchers in academics and commerce have the option to upload a list of authorship identities from various projects. This list will be fed to the disambiguation process [2] in the backend which can correct low quality developer identity data derived from various online tools. Once the correction is complete, the tool will export the corrected list of developer IDs to the user via email.

V. Planned Evaluation Studies

The approach for evaluation studies starts with the recognition of two user bases: those looking to utilize developer profiles and those strictly seeking identity disambiguation capabilities.

Developer Profiles: To evaluate the tool's capability and effectiveness in regards to the developer profiles it generates, a modular approach has been designed. The initial study includes a smaller user base consisting of co-workers, peers, and others within a close circle who have shown interest and are already signed up as users in the early stages of the tool. Since this user base provides easy contact, the intent is to perform either in-person or online interviews to obtain initial reactions, feedback, and evaluation of the tool. Some of the insights we hope to gain include:

- As a new user, how difficult was it to navigate and get full use out of the tool?
- What features would you like the tool to have that it does not currently have?
- In what scenarios can you anticipate using this tool?
- Did you get what you wanted out of the tool?

Once this initial wave of feedback has resonated in the form of new features, deletions, or any modifications with the tool, a second set of evaluation has been planned to better gauge effectiveness. With this second study, the goal is to bring competitors, such as GitHub and LinkedIn, into the picture to compare how our tool holds up against the competition. The study revolves around providing users with a "task" (e.g. find a developer to work on someone's Javascript project) and evaluating which platform provides the "best" suggestion for each, individual user. From there, we can gauge what our tool lacks in terms of developer profile information, what users look for when comparing developers, etc. We plan to investigate the utility of DRE for badging as well. First, we will create a set of our own (impact) badges and compare them to GitHub's current badges to answer questions such as - "Whose badge did you find more useful? Whose pull request are you more willing to merge based on GitHub's badge vs. our badge?".

Identity Disambiguation: We plan to measure the precision of the identity disambiguation tool suggestions by counting how many of the suggested author IDs are accepted as correct by the developer. We also plan to measure how the accuracy of the disambiguation approach increases with more and more data gathered. Finally, we will consider ways to address ethical (a barrier to entry for underrepresented groups), privacy (email harvesting), and societal (artificially boosting the measure) issues that may arise in the course of the tool usage.

[1] https://en.wikipedia.org/wiki/Erdos_number

References

[1] Y. Ma, C. Bogart, S. Amreen, R. Zaretzki and A. Mockus "World of Code: An Infrastructure for Mining the Universe of Open Source VCS Data," in Proceedings of the International Conference for Mining Software Repositories. Montreal, May 2019.

[2] S. Amreen, R. Zaretzki and A. Mockus, C. Bogart and Y. Zhang "ALFAA: Active Learning Fingerprint Based Anti-Aliasing for Correcting Developer Identity Errors in Version Control Data," arXiv preprint arXiv:1901.03363, 2019.

[3] A. Mockus and J. D. Herbsleb, "Expertise browser: a quantitative approach to identifying expertise," in IEEE, Proceedings of the 24th International Conference on Software Engineering, 2002, pp. 503–512.

[4] R. Milewicz, G. Pinto and P. Rodeghero, "Characterizing the Roles of Contributors in Open-source Scientific Software Projects" in Proceedings of the International Conference for Mining Software Repositories. Montreal, May 2019.

[5] R. Venkataramani, A. Gupta, A. Asadullah, B. Muddu and V. Bhat "Discovery of technical expertise from open source code repositories," in Proceedings of the 22nd International Conference on World Wide Web, pp. 97–98, 2013.

[6] A. S. Badashian, A. Esteki, A. Gholipour, A. Hindle, and E. Stroulia, "Involvement, contribution and influence in GitHub and stack overflow," Proceedings of 24th Annual International Conference on Computer Science and Software Engineering, vol. 2, pp. 19–33, 2014.

[7] A. Bosu and J. C. Carver,"Impact of developer reputation on code review outcomes in oss projects: An empirical investigation," Proceedings of the 8th ACM/IEEE international symposium on empirical software engineering and measurement, pp. 33, 2014.