Developing an H-Index for OSS Developers

Andrea Capiluppi
Brunel University
London, United Kingdom
andrea.capiluppi@brunel.ac.uk

Alexander Serebrenik
MDSE, Eindhoven University of Technology
Eindhoven, The Netherlands
a.serebrenik@tue.nl

Ahmmad Youssef
ACE, University of East London
London, United Kingdom
ahmed.ali.youssef@hotmail.com

***Abstract*—The public data available in Open Source Software (OSS) repositories has been used for many practical reasons: detecting community structures; identifying key roles among developers; understanding software quality; predicting the arousal of bugs in large OSS systems, and so on; but also to formulate and validate new metrics and proof-of-concepts on general, non-OSS specific, software engineering aspects.**

One of the results that has not emerged yet from the analysis of OSS repositories is how to help the "career advancement" of developers: given the available data on products and processes used in OSS development, it should be possible to produce measurements to identify and describe a developer, that could be used externally as a measure of recognition and experience.

This paper builds on top of the h-index, used in academic contexts, and which is used to determine the recognition of a researcher among her peers. By creating similar indices for OSS (or any) developers, this work could help defining a baseline for measuring and comparing the contributions of OSS developers in an objective, open and reproducible way.

I. Introduction

Being part of an OSS community is becoming an increasingly common experience among software developers, for several reasons: to learn a new programming language [1]; to study the OSS processes or products as part of an undergraduate or postgraduate programme [2]–[4]; to get involved and interact with a stimulating team, a distributed development environment with expert peers developing a system or system of systems [1]; or more commonly to peruse more altruistic (or "intrinsic") social needs and objectives [5], [6]. It has also become clear that these motivations are different from the more technically- or individually-oriented (i.e., "extrinsic") motivations of paid software engineers [7].

One of the less studied reasons for OSS developers to get involved with an OSS project is career advancement: developers and active OSS volunteers are motivated by showing off their skills, to be judged by potential employers [8] as developers, project managers, testers or community leaders. Their experience in OSS projects needs to be quantifiable and objective, so to be added to their profile: several respondents in a recent survey [9] noted that OSS experiences reflected in GitHub profiles now act as a portfolio of work and factor into the hiring process at many companies.

The body of recent empirical work on OSS is so varied and ample that it is difficult to summarize the main findings in clear and unequivocal terms: one way of understanding the current related literature is dividing it in studies "on OSS systems" [10], [11] and studies "with OSS systems" [12]. Specifically analysing the behavior of OSS developers, several works have highlighted the presence of core and additional developers, their level of engagement [13], the effects of territoriality [14], and even frameworks for comparing and contrasting the processes of different communities [15]. Such research demonstrates that OSS projects are based on effort that should be acknowledged, and that this effort varies dramatically among individuals in some cases. Therefore, it should be possible to formulate an index (or set of indexes) clearly conveying the information on individual developers, to be used as an external metric by hiring managers when recruiting new developers[1]. Existing external indicators (e.g., the *Kudos Ranks* in the Ohloh community[2]) are reputation-based: a higher rank is only achieved when other members of the same community will vote for an individual, not for objectively measured merits on specific projects.

An example of an external metric recently adopted by academics and recruiters is the *h-index* [16]: from an academic standpoint, this index measures both the productivity and the external recognition of a researcher, by collecting the number of citations of all her publications: having (at least) N publications with (at least) N citations each will produce an h-index of N. Although flaws have been identified in how the h-index is evaluated and threats to its validity posed [17], the index is widely used, for instance in identifying successful candidates for academic positions [18].

This paper studies how an h-index could be designed to characterize the activity of OSS developers: it does so by using the results from the literature and publicly available data sources, and in order to try and answer three needs:

1) **need to give a value to OSS experiences**: developers spending time in an OSS project should be able to "claim their time" from their experience;
2) **need for objective measures**: given the openness of the data, the process of extrapolating a measurement of an OSS experience should be based on objective criteria, and measurable and reproducible steps; and

[1]Such index would not be sufficient: large IT companies lately ask candidates to sit a 'BrainBench' exam before the face-to-face interview.
[2]http://www.ohloh.net/people/rankings?query=&sort=kudo_position

3) **need for a means to distinguish between developers**: the produced measurements should be used to compare and contrast experiences and skills, across communities, application domains and responsibilities.

II. Rationale

The h-index was designed with the typical highly-skewed distribution in mind: the majority of researchers has few papers with a large number of citations, and a larger number of publications with few citations. This pattern has also been observed in various aspects of the OSS development [19]–[22], and it is especially visible when developers work on different projects at the same time (see Figure 1 where the activity of developer D1 is summarized). Paraphrasing the h-index, from Figure 1 D1 has an indicator of 11: s/he has contributed more than 11 commits in at least 11 projects.

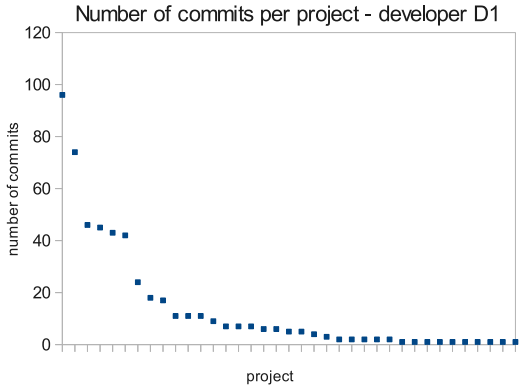

Figure 1. Skewed activity of developer D1 over several projects

Such measurement, although easy to evaluate and straightforward to understand, would not be very helpful: it tells little about the projects participated, the team dynamics, the usefulness of the system or the role of the developer in each. If used for career advancement, a hiring manager would need quantitative and qualitative data, whereas this index:

1) does not account for the number of *contributors* in a project. An OSS h-index should keep track of who's working in projects with large communities. Participating in a large project would prove certain technical skills, but it would be difficult to discern those developers who "free-ride" its reputation;
2) does not account for how *useful* the projects were (i.e., a "read" publication), or if they were used (i.e., a "cited" publication) by other projects. An OSS h-index should track how a project is used (i.e., downloaded) or needed (i.e., linked) by others;
3) does not consider the *length of engagement* of individuals: an experienced developer has a more consistent and lengthy commitment than an apprentice. The original h-index is also formulated for evaluating researchers in the same stage of their careers [16];
4) does not account for the roles of a developer in the projects (developer, owner, tester, or any other roles).

III. Designing an h-index for OSS developers

The availability of open data and shared repositories could have an immediate effect on the visibility of OSS developers, as long as an accepted framework for evaluating an index (or several indexes) was discussed and validated. Below one of the possible approaches for operationalization is discussed, with examples and results taken from two available repositories (GNOME[3] and GoogleCode[4]).

Quantitatively, the framework to design an index to describe OSS developers should be:

- *open*: it should be possible to evaluate such an index by using openly available data, or by parsing existing data sources freely available on the web;
- *understandable*: the index should have a direct link with the measured characteristics, and based on an ordinal or a scale factor;
- *objective*: the index should be transparent in how evaluates the characteristics under evaluation;
- *reproducible*: the availability of open data should be an advantage, and allow any interested party to replicate the index for individual developers, considering different repositories or specific OSS communities.

Qualitatively, the framework should

1) discuss the contributions and the status of the developer: hiring managers are surely interested in whether the individual was the owner or a committer in a project, the features she worked on, the productivity compared to others and so on;
2) describe the project(s), its goals and its scenarios of use. A small, contained project with a small audience is as valuable as a large and multi-developed project;
3) describe the team dynamics: from the perspective of a hiring manager, it is relevant to understand whether a candidate has experience in working together with other developers on the same features.

The original h-index does not clarify the background of the publications, or whether an outlet is more or less difficult to publish in: it does, however, tie to a specific research field ("Computing", "Medicine", etc). Similarly, the OSS h-index could be broken down across skill-sets (e.g., domain or programming language): a small internet company is more interested in developers having previous experience in creating web applications, rather than training a candidate with a different (albeit impressive) skill set.

[3]http://www.gnome.org

[4]http://code.google.com/: dumps are hosted by the FLOSSMole project.

A. An h-index for the committers status

As visible in Figure 1, the same developer participates unevenly in the production of various projects. This aspect should be captured quantitatively: assuming that a project has N developers, and considering the top 10 committers (in terms of number of commits), the h-index $h_{A,committer} = M$ would mean that "*in at least M projects the developer A is in the top 10 committers*". Differently from the absolute measures of Figure 1, this index would produce a measurement of how many projects are participated by one single developer, and how she scores in comparison to other committers.

To exemplify the usage of such index, the whole GoogleCode repository was analysed, and the projects with the most committers identified. Investigating the "go" project (http://code.google.com/p/go/, with 186 overall participants), the top committer (say, *Dev1* for privacy) is found to be also contributing in 8 other projects: since he's also the top committer in all the participated projects, this could be reflected in the index by appending decimals to the index. For Dev1, the value of $h_{Dev1,committer} = 9.9$, meaning 9 projects where he participates as one of the 10 committers, and being the most prolific committer (or the likely owner) in 9 of them (shown as decimals). Similarly, Dev 8 participates in 4 projects, being in the top-10 committers in 3 projects, and the most prolific in 2, hence producing an index of 3.2.

In the evaluation of such index, it would be even possible to use single "releases" instead of full projects, by claiming that "*a developer worked as a top-committer in N releases*" of the same project, hence reinforcing the similitude with related publications by the same authors.

Table I
THE $h_{A,committer}$ INDEX PER AUTHOR IN THE "GO" PROJECT

	Top Ranking in the "Go" project		
Developer	projects participated	top ten in	index
Dev1	9	9	9.9
Dev2	2	2	2.0
Dev3	2	2	2.0
Dev4	2	2	2.0
Dev5	3	3	3.0
Dev6	3	3	3.1
Dev7	3	3	3.1
Dev8	4	3	3.2
Dev9	3	2	2.1
Dev10	3	2	2.0

B. An h-index for the community sizes

The second aspect to highlight in an index would be the community sizes: paraphrasing the h-index, $h_{A,community} = N$ means that the "*developer A has worked in at least N communities containing at least N contributors*" (owners and/or committers). From Table I, Dev1 participates in 9 projects, having $\{186, 108, 4, 3, 2, 2, 2, 1, 1\}$ developers, respectively. This produces a $h_{Dev1,community} = 3$: he participates in at least 3 projects with at least 3 other developers each. Conceptually, this second index would help isolating developers working "alone" in many projects.

Considering GNOME, the developer with the highest $h_{A,community}$ value (i.e., 129) is Kjartan Maraas, the most prolific unpaid contributor in the GNOME project [23], while Carl Worth (who predominantly focuses on one project, according to [23]) has a value of 5. Furthermore, we have observed that GNOME contributors with high $h_{A,community}$ values are also active in translation, e.g., the contributor responsible for Swedish translations has the h-index of 128, and the core maintainer of the Arabic localization scores 117. This is to be expected since translation activities in GNOME are supported by a common translation collaboration tool (Vertimus).

C. One h-index for the "usage" and one for the "citations"

One of the most important factors of any software product is whether that is useful or not, for either its users, or for the developers of other systems, who incorporate whole or parts of that system into theirs: these two attributes could be used to formulate an index of the project "importance".

Regarding the first aspect, we considered the developers, the participated projects and the cumulated number of downloads of each project. Table II shows an example of it: the numbers are based on Dev X and Dev Y participating in the four most participated in projects of GoogleCode, creating the $h_{A,downloads}$ index for downloads. Since the downloads are of the order of magnitude of the thousands, a $\log_{10}$ was used: as a result of this evaluation, both Dev X and Dev Y have an index of 3, since both are engaged in at least 3 projects with at least 10^3 downloads each. As above, this could be also evaluated at the release level.

Table II
EVALUATING THE $h_{A,downloads}$ INDEX PER AUTHOR

project	Downloads	$\log_{10}$	$DevX$	$DevY$
google-web-toolkit	8,601,795	6.93		✓
closure-compiler	78,491	4.92	✓	✓
closure-library	18,247	4.26	✓	✓
plovr	1,675	3.22	✓	✓
		$h_{A,d}$	3	3

Regarding the second aspect, one would need to define how a project score in terms of its "citations" in other systems: this requires more work, since all of the OSS systems have to be investigated to establish relationship of use between each other. Some work has been already proposed to uniquely identify a component, or a generic entity in a large pool of available data [24].

IV. THREATS TO VALIDITY

Although an initial proposal, formulating these indexes following the original h-index generates several threats to validity. *First*, scientific publications have a specific form, while software is continuously evolving. One way to address

this problem would be to focus on official releases. *Second*, scientific publications are being reviewed, i.e., there is some kind of quality control. Regarding source code, it is not very clear what has been reviewed and what not. One possibility here would be to think about "signed off by" of Git or similar mechanisms. *Third*, "having participated to a large and successful project" has to be properly delimited: the areas worked on should be identified, the length of the commitment, the releases worked on and the kind of contribution (e.g., translation or coding). Similar issues appear for the original h-index, where large groups of coauthors are acknowledged of the same publication. *Fourth*, in the same way as Évariste Galois' h-index is 2 despite his huge impact on modern mathematics, OSS h-indices should be applied with care: e.g., $h_{A,committer}$ favors involvement in small (in terms of number of committers) projects.

V. Conclusion

This paper tackles a relatively new research question, by designing and implementing a first draft of an index that could be used to describe the experience of an OSS developer on one or many projects. The rationale of adapting the h-index, used in academia, is based on similar patterns observed both in the number of citations per paper, and the typical involvement of developers in multiple OSS projects. By formulating indexes that reflect both the number of projects participated, and the activities run in the various projects (engagement, structure of the communities, number of downloads, reuse in other project), it could be possible to capture various aspects of the experience in OSS projects, while preserving the knowledge of the inherent skewness of the distribution when participating in multiple projects.

References

[1] A. Hars and S. Ou, "Working for free? motivations for participating in open-source projects," *Int. J. Elect. Commerce*, vol. 6, pp. 25–39, 2002.

[2] D. Megías, J. Serra, and R. Macau, "An international master programme in free software in the European higher education space," in *Open Source Systems*, 2005, pp. 349–352.

[3] S. Sowe and I. Stamelos, "Involving software engineering students in open source software projects: Experiences from a pilot study," *Journal of Information Systems Education*, vol. 365, no. 4, pp. 349–352, 2007.

[4] V. Goduguluri, T. Kilamo, and I. Hammouda, "Kommgame: A reputation environment for teaching open source software," in *OSS*, ser. IFIP Publications, S. A. Hissam, B. Russo, M. G. de Mendonça Neto, and F. Kon, Eds., vol. 365. Springer, 2011, pp. 312–315.

[5] A. Bonaccorsi and C. Rossi, "Why open source software can succeed," *Research Policy*, vol. 32, no. 7, pp. 1243–1258, 2003.

[6] G. Hertel, S. Niedner, and S. Herrmann, "Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel," *Res. Policy*, vol. 32, no. 7, pp. 1159–1177, 2003.

[7] T. Hall, N. Baddoo, S. Beecham, H. Robinson, and H. Sharp, "A systematic review of theory use in studies investigating the motivations of software engineers," *ACM Trans. Softw. Eng. Meth.*, vol. 18, no. 3, 2009.

[8] J. Lerner and J. Triole, "The simple economics of open source," National Bureau of Economic Research, Working Paper 7600, March 2000.

[9] L. A. Dabbish, H. C. Stuart, J. Tsay, and J. D. Herbsleb, "Social coding in GitHub: transparency and collaboration in an open software repository," in *CSCW*, S. E. Poltrock, C. Simone, J. Grudin, G. Mark, and J. Riedl, Eds. ACM, 2012, pp. 1277–1286.

[10] M. Conklin, J. Howison, and K. Crowston, "Collaboration using ossmole: a repository of floss data and analyses," in *MSR*, 2005, pp. 1–5.

[11] H. Schackmann and H. Lichter, "Evaluating process quality in gnome based on change request data," in *Proc of the 2009 6th IEEE International Working Conference on Mining Software Repositories*, ser. MSR '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 95–98.

[12] J. Ekanayake, J. Tappolet, H. Gall, and A. Bernstein, "Tracking concept drift of software projects using defect prediction quality," in *MSR*, M. W. Godfrey and J. Whitehead, Eds. IEEE, 2009, pp. 51–60.

[13] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and Mozilla," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 3, pp. 309–346, 2002.

[14] G. Robles, J. M. Gonzalez-Barahona, and J. J. Merelo, "Beyond source code: the importance of other artifacts in software development (a case study)," *J. Syst. Softw.*, vol. 79, pp. 1233–1248, September 2006.

[15] M. Goeminne and T. Mens, "A framework for analysing and visualising open source software ecosystems," in *EVOL/IWPSE*, A. Capiluppi, A. Cleve, and N. Moha, Eds. ACM, 2010, pp. 42–47.

[16] J. E. Hirsch, "An index to quantify an individual's scientific research output that takes into account the effect of multiple coauthorship," *Scientometrics*, vol. 85, pp. 741–754, December 2010.

[17] C.-T. Zhang, "Relationship of the h-index, g-index, and e-index," *J. Am. Soc. Inf. Sci. Technol.*, vol. 61, pp. 625–628, March 2010.

[18] L. Bornmann and H.-D. Daniel, "Does the h-index for ranking of scientists really work?" *Scientometrics*, vol. 65, no. 3, pp. 391–392, 2005.

[19] C. Bird, N. Nagappan, B. Murphy, H. Gall, and P. T. Devanbu, "Don't touch my code!: examining the effects of ownership on software quality," in *SIGSOFT FSE*, T. Gyimóthy and A. Zeller, Eds. ACM, 2011, pp. 4–14.

[20] G. Madey, V. Freeh, and R. Tynan, "The Open Source Software development phenomenon: An analysis based on social network theory," in *Proc of the Eighth Americas Conf on Inf Syst*, 2002, pp. 1806–1815.

[21] P. Louridas, D. Spinellis, and V. Vlachos, "Power laws in software," *ACM Trans. Softw. Eng. Methodol.*, vol. 18, pp. 2:1–2:26, October 2008.

[22] A. Serebrenik and M. van den Brand, "Theil index for aggregation of software metrics values," in *ICSM*. IEEE, 2010, pp. 1–9.

[23] D. Neary and V. David. (2010) The GNOME census: Who writes GNOME? [Online]. Available: http://blogs.gnome.org/bolsh/files/2010/07/GNOME-Census.pdf

[24] J. Davies, D. M. German, M. W. Godfrey, and A. Hindle, "Software bertillonage: finding the provenance of an entity," in *MSR*, 2011, pp. 183–192.