Measuring Open Source Software Impact

Emergent Research Forum (ERF)

Vinod K. Ahuja
University of Nebraska at Omaha
vahuja@unomaha.edu

Matt Germonprez
University of Nebraska at Omaha
mgermonprez@unomaha.edu

Abstract

Open source software foundations and communities often want to know the impact of their software. This impact can be understood in a variety of ways and in this paper we explore impact through the interdependencies of open source software. In this, open source software is dependent on components created upstream and open source software is used in components downstream – impact within an open source supply chain. This paper proposes an index (called the V-index) through which impact of an open source software, as used in downstream components, can be measured. This index is developed using the open database libraires.io, which provides the dependencies of open source software distributed through various package managers. The proposed index helps measure the impact of an open source software as part of its use within an open source supply chain.

Keywords (Required)

Open Source Software Impact, Software Dependencies, V-index

Introduction

Much of the technology we use today is developed with the use of open source software. Open source software is not only used in developing software in the IT industry but it is penetrating other industries including automotive, industrial, and logistics.

To date, numerous open source communities have formed to focus on specific technology problems and community members contribute to the solution of those problems and the development of related software. With the freedom and accessibility of open source software, software development is becoming dependent on such software (Spinellis and Szyperski 2004). For example, in the field of data science we collect data, analyze data, and present the results. For collecting data, we import one open source library that connects to a database and imports data for us in the desired format. We import open source libraries to format the collected data. We import open source libraries to analyze and make sense of data. Finally, once the data is analyzed, open source libraries are used for its presentation. So, our own software is a collection of open source software components.

Today, software development is nearly impossible without the use of these interdependent components. These interdependencies have such a strong impact, that software often breaks if one dependent open source software library malfunctions. This was well observed in an example when an open source software contributor deleted his 11 lines of code he contributed to the NPM open source community and thereby halted many websites dependent on those 11 lines (Collins 2016).

Related to these interdependencies, the impact of open source software, as used within a software ecosystem[1], can be observed. In literature, we find work which speaks to the impact of software on business and society but no one has measured the impact of these interdependent software libraries, leading to our research question:

[1] A software ecosystem is "a collection of software projects which are developed and evolve together in the same environment. The environment is usually a large company, an open-source community, or a research group. It is even possible for multiple organizations to collaborate and develop software in a common ecosystem" (Lungu et al. 2010).

RQ: How can we measure open source software impact as related to open source software interdependencies?

Though this research we study open source software impact as this will help in understanding the larger proposition associated with open source software value (CHAOSS 2017). Further, we explore how to quantify the impact of open source software within a software ecosystem as one piece of software is represented via its upstream and downstream interdependencies[2].

Related Literature

Impact is evaluated in many fields. In marketing we have 'advertising impact', in finance we have 'market impact', in social science we have 'social impact', in academe we have 'scholar impact', and in IT we have 'software change impact' (Hearn and Buffardi 2016). Everyone measures impact in different ways and there is no single agreed method through which impact can be measured. The various methods (i.e., qualitative and quantitative) for measuring impact are determined by the context in which impact is measured, by the purpose for which impact is evaluated, and by the kind of data that is available. (Robert Chambers et al. 2009).

Specific to this study, researchers have measured the impact of open source software in various ways. As samples, Borges, Hora, & Valente (2016) have measured the impact by age, commits, contributors, forks, programing language, application domain, and release of new features on the popularity of open source software. Further, Miller (2016) used a number of stars of a GitHub repository to calculate the impact of an open source software. However, monitoring a project does not necessarily mean use, and subsequent impact of the project. So, having stars, watcher, and followers can be a proxy of impact but not the actual impact of the project. Table 1 provides a summary.

Measure of Software Impact	Definition	Sample References
Popularity	Age, commits, forks, contributors, programing language, new features, and application domain as impacting the popularity of software measured in terms of a number of stars.	Borges, Hora, & Valente (2016) found in half of the repositories gain 53% increase in a number of stars after a major bug release.
Stars	If an organization or an individual has N number of projects and each project has N number of stars then their impact is N and is called gh-impact.	Miller (2016) found that Mozilla has a gh-impact of 95 meaning 95 of their projects have at least 95 stars to each project.

Table 1. Summary of Sample Open Source Project Related Impact Studies

In our research, we explore how to measure the impact of open source software in terms of downstream dependencies – proposing a V-index (Value Index) similar to H-index – which measures the impact of open source software similar to the impact of a scholar in an academia. As in academe, researchers publish articles and those articles are cited. The H-index helps answer the question of how to determine researcher impact. We believe that such a comparison is helpful in quantifying software impact within a larger software ecosystem (Hirsch 2005).

[2] Think of upstream and downstream interdependencies in a river metaphor. If something is upstream to my position in a river, I am reliant on it. If something in downstream to my position in a river, it is reliant on me. In software, upstream dependencies are the pieces of software that I rely on in my development work. In software, downstream dependencies are the pieces of software the rely on my development work.

Impact of a scholar is measured in number of ways like number of publications, number of citations, citations over a period of time. Hirsch (2005) proposed H-Index which accounted both the number of publications and number of citations of a scholar in their academic field. If a scholar has N number of publication and each publication has at least N number of citations than the H-impact of that scholar is N. In a similar way we developed V-Index to measure the impact of open source software through their dependencies. The detailed methodology of V-Index calculation follows.

Methodology

We propose to measure open source software impact via the development of quantitative methods. We are aware that there is commercial software that keeps track of source code dependencies of various software libraries. Such commercial software maps dependencies in version control systems. This helps to keep track of interdependent libraries — and if any change or upgrade is made in a library, the impact on entire software can be evaluated (Arnold and Bohner 1996). In this, we can measure impacts on one particular piece software but not necessarily the impact that one piece of software has on a software ecosystem.

In response, we propose a method to address our research question: *how can we measure open source software impact as related to open source software interdependencies?* We develop a V-index through which we calculate the impact of an open source software in form of dependency that software has over other software.

Data

Librairies.io[3] provides an extensive database for open source software library dependencies. They classify this data by 33 open source package managers through which open source software is distributed. Libraries.io keeps a record of all the open source software which is distributed through these package managers. Libraries.io is freely accessible through their API and has published the entire database in 7 CSV files of the total size of 7.1GB containing 311 million rows. The data contains information of all the open source software with their various versions and using various libraries they depend on in each version.

V-index

There are two type of dependencies for software. First, upstream dependency, whereby the software in question depends on various other software. Second, downstream dependencies, whereby other software depends on the software in question (Haenni et al. 2014). The proposed V-index is formed similar to H-index whereby the impact of an open source software is measured in the form of downstream dependencies.

The V-index is calculated in two steps. First, downstream dependencies of a software in question are extracted from the libriaires.io database. This forms the first order dependencies as to how many pieces of software are dependent on the software in question – similar to how many papers cite a paper in question. Second, each extracted downstream dependency is further evaluated and their own downstream dependencies are recorded. These form the second order dependencies – similar to a second order set of citations stemming from a paper in question. As one piece of software has N number of dependencies in the first order and each dependency in the first order has their own N number of dependencies in second order then the V-index of that software is N. As explained in the example in figure 1. V-index is further elaborated in three different scenarios presented in Table 2.

[3] https://libraries.io/

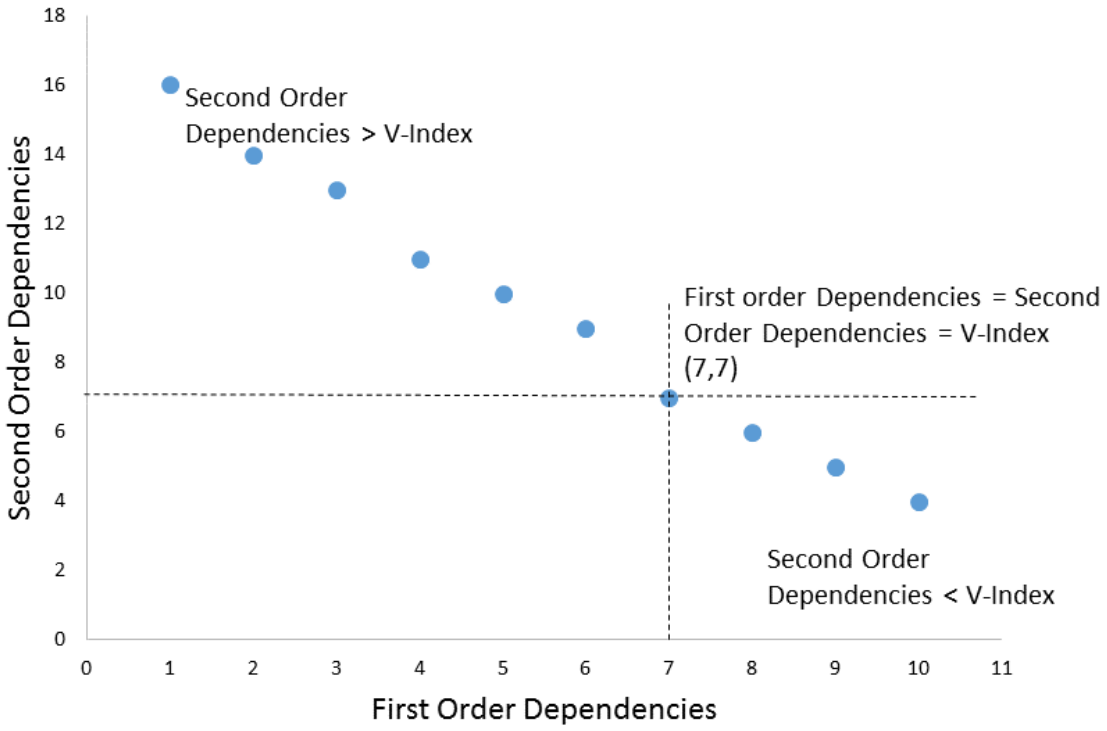

Figure 1. V-index where Software X has 10 dependencies in the first order. However, only 7 of first order dependencies have 7 dependencies of their own (second order). So the V-index of Software X is 7.

Scenario A	
First Order Dependencies	**Second Order Dependencies**
Dependency 1	0
Dependency 2	0
Dependency 3	0
Dependency 4	0

Project A has four projects that depend on it. No other project depends on these projects. The V-Index of Project A is zero because zero first order dependencies have more than zero second order dependencies.

Scenario B	
First Order dependencies	**Second Order Dependencies**
Dependency 1	4
Dependency 2	4
Dependency 3	4
Dependency 4	4

Project B has four projects that depend on it. Each of these projects has 4 projects that depend on them. The V-Index of Project B is four because each of the four first order dependencies have at least four second order dependencies.

Scenario C	
First Order dependencies	**Second Order Dependencies**
Dependency 1	40

Project C has one project that depends on it. This project has 40 projects depend on it. The V-Index of Project C is one because it has one first order dependency that has at least one second order dependency.

Table 2. Scenarios elaborating V-index

Conclusion

This research contributes to the literature by providing methods through which impact of open source software can be evaluated via dependencies. Further research can also be done to explore how the V-index can be increased. A relationship between V-Index and other open source measures like stars, forks, commits, bugs, closed issues, and pull request can be evaluated.

This research also contributes to the open source community, CHAOSS, by providing a method through which health of open source projects can be measured in terms of their dependencies. A list of V-Index of various projects can be created to measuring the health in comparison to other projects. With this comparison various open source foundations and communities can evaluate the impact of their projects

on their ecosystem. Foundations and communities knowing the impact of their projects can realign their efforts to the high impact projects by maintaining and improving their quality and also by taking required measures to improve the low impact projects.

Acknowledgements

This project was supported by grants from Mozilla and the Alfred P. Sloan Foundation regarding Open Source Health and Sustainability.

ORCID

Vinod K. Ahuja orcid.org/0000-0002-9227-2921

Matt Germonprez orcid.org/0000-0003-2326-5901

References

Arnold, R., and Bohner, S. 1996. "Software Change Impact Analysis," July. (http://www.wiley.com/WileyCDA/WileyTitle/productCd-0818673842,miniSiteCd-IEEE_CS2.html, accessed February 19, 2018).

Borges, H., Hora, A., and Valente, M. T. 2016. "Understanding the Factors That Impact the Popularity of GitHub Repositories," *ArXiv:1606.04984 [Cs]*. (http://arxiv.org/abs/1606.04984).

CHAOSS. 2017. (https://wiki.linuxfoundation.org/chaoss/metrics, accessed October 10, 2017).

Collins, K. 2016. "How One Programmer Broke the Internet by Deleting a Tiny Piece of Code," *Quartz*, March 27. (https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code/, accessed January 20, 2018).

Haenni, N., Lungu, M., Schwarz, N., and Nierstrasz, O. 2014. "A Quantitative Analysis of Developer Information Needs in Software Ecosystems," in *Proceedings of the 2014 European Conference on Software Architecture Workshops*, ECSAW '14, New York, NY, USA: ACM, 12:1–12:6. (https://doi.org/10.1145/2642803.2642815).

Hearn, S., and Buffardi, A. L. 2016. "What Is Impact?," *ODI*, February. (https://www.odi.org/publications/10326-what-impact, accessed January 23, 2018).

Hirsch, J. E. 2005. "An Index to Quantify an Individual's Scientific Research Output," *Proceedings of the National Academy of Sciences* (102:46), pp. 16569–16572. (https://doi.org/10.1073/pnas.0507655102).

Ireland, T., MacDonald, K., and Stirling, P. 2012. "The H-Index: What Is It, How Do We Determine It, and How Can We Keep up with It," *Science and the Internet*, pp. 237–247.

Lungu, M., Lanza, M., Gîrba, T., and Robbes, R. 2010. "The Small Project Observatory: Visualizing Software Ecosystems," *Science of Computer Programming* (75:4), Experimental Software and Toolkits (EST 3): A Special Issue of the Workshop on Academic Software Development Tools and Techniques (WASDeTT 2008), pp. 264–275. (https://doi.org/10.1016/j.scico.2009.09.004).

Miller, I. D. 2016. "Gh-Impact - Questions and Answers - A Brief on Gh-Impact Methods," *Gh-Impact*, August. (http://www.gh-impact.com/reports/questions-and-answers.html, accessed February 19, 2018).

Robert Chambers, Dean Karlan, Martin Ravallion, and Patricia Rogers. 2009. "Designing Impact Evaluations: Different Perspectives," July. (http://www.3ieimpact.org/en/publications/working-papers/working-paper-4/, accessed February 17, 2018).

Spinellis, D., and Szyperski, C. 2004. "How Is Open Source Affecting Software Development?," *IEEE Software; Los Alamitos* (21:1), pp. 28–33. (http://dx.doi.org/10.1109/MS.2004.1259204).