# Correlation Analysis of Popularity and Interoperability in Open Source Projects

Fatima Al Shamsi
*Abu Dhabi Systems and Information Centre*
Abu Dhabi, UAE
Fatima.AlShamsi@adsic.abudhabi.ae

Sarah Bamatraf, Talal Rahwan, Zeyar Aung, and Davor Svetinovic
*Khalifa University of Science and Technology*
Abu Dhabi, UAE
{sarah.bamatraf, talal.rahwan, zeyar.aung, davor.svetinovic}@ku.ac.ae

***Abstract*—In order to examine the success of open source projects, we analyzed 252,008 projects in the SourceForge database. We restricted our study to projects that are written in the top ranked programming languages. We measured the correlation between popularity, interoperability, productivity and the success rate of each project. The developers' contribution to the project was also examined in terms of the developer's team size, and the success in terms of the total number of committers (contributors). Our results indicate that using multiple programming languages requires additional team members. Also, there is a significant increase in the functionality of the developed project. The results also demonstrated that there is a positive correlation between the development team size, and the total number of committers.**

***Index Terms*—open-source software, programming languages, data mining**

## I. Introduction

In spite of decades of studies, what makes a programming language used by people of all skill levels remains elusive. Although a considerable number of programming languages is still used by the community, the reasons behind preferring specific languages is a subject of debate [1]. This paper examines the success of different open-source projects, especifically those written in the top-ranked programming languages, to measure their success rate. The concept of open source software is increasingly gaining attention worldwide. Open source software is usually developed by programmers spread across the whole world. They take pride in contributing towards the development of such software projects [2]. Open source projects have a significant impact on the software business globally. In a 2009 survey, it was found that 24% of all responding enterprises were using or implementing open source software [3].

In our study, different correlations are observed that associate software development and common programming languages with the success of open source projects in terms of the developers team size, and the total number of committers. Software development is a closely-coupled activity, which often involves tight integration and inter-dependencies between modules. Thus, it normally requires a substantial amount of coordination and communication between its developers [4]. The term *interoperability* will be used to refer to the process of developing software projects using multiple programming languages.

To date, the number of papers that study the popularity, interoperability, and impact of diverse programming languages is limited. Hence, there is an urgent need for a comprehensive analysis of the mentioned factors, by analyzing the projects that are written in several programming languages. Open source practices and tools have been remarkably successful. But, in several areas there are still opportunities for improvement. These areas include supporting collaboration among developers, supporting new developers, supporting the broader end-user community including non-developers, and understanding end-users' requirements [5].

For the purpose of our study, 494,158 SourceForge code repositories were analyzed. The number of committers, as well as the total number of developers in each given project were investigated. Upon these information, our hypotheses was developed in a structured manners, based on the popularity and interoperability of programming languages.

Our research is centered around two research hypotheses that are designed to find correlation between popularity, interoperability, and the impact of programming languages in open source projects.

**H1**: *Being written in more than five programming language increases the productivity and the success rate of projects.* We verify whether increasing the level of interoperability contributes to the success of the projects, or the contrary as writing in multiple programming languages requires larger team sizes of skillful developers.

**H2**: *Being written in top-ranked programming languages increases the potential of project success.* We investigate the correlation between the top 10 programming languages in SourceForge and the success of the projects, while considering metrics such as committers, and the size of the development team.

## II. Related Work

The history, the fundamentals, the trends, and the advancement of programming languages have been studied by several research works. Open source users have conventionally been thought of as similar to developers [6]. The findings further suggest that a reflexive project tends to be more popular among users than those intended for a more general audience.

It was found that files modified by a developer can be used to give an approximation of developer expertise [7]. The new

978-1-5386-6673-9/18/$31.00 ©2018 IEEE

users can then find out who asks for advice while modifying certain codes. Code contributors have the ability to investigate who to suggest as reviewers for the code. This process can facilitate the identification of the best candidates to fix any given problem. Other research has focused on the different versions of a software system, in order to analyze the source codes of these versions. Also, documentation and other meta-data can reveal regulations and anomalies in the development process of the system [8].

The Concurrent Versions System (CVS) repositories of 9,999 open source projects that are hosted on the SourceForge website were examined. The 10 most popular programming languages have been tested, and it was found that a programming languages is an essential factor when determining the rate at which source code is written, even after accounting for the variations between programmers and projects [9].

Feitelson and Heller [10] wanted to determine a success criterion for open source software projects. To this end, they analyzed 122,205 projects in the SourceForge database, and found that there were 80,597 projects with no downloads at all. The remaining 41,608 projects were categorized into three categories: super projects (those with over 1.1 million downloads; this category contained 85 projects), successful project (those with more than 1,680 downloads each; this category contained 10,000 projects), and struggling projects (those with less than 1,680 downloads).

As for the team size, Grottke et al. [11] noticed that a larger team might increase the average pending time. Higher team dispersion also affects the pending time, since identifying the developer that is responsible for a certain failure become more complicated. As for the team experience, more experience increases the efficiency of handling failures, since members get more accustomed when dealing with failures. After processing numerous failure reports, the authors concluded that team members typically know who is responsible for each given task of the coding part, and they are also able to solve problems quickly.

The success factors of 134,549 open source software projects that are available at SourceForge.net were identified by using the data mining 2-itemset association rule [12]. Data from each projects such as potential audience, database environment, description, development, file name, file size, downloads, topic and programming language were all collected and stored into a database to be analyzed. The result of this study demonstrated six success factors that may be applied by project initiators and developers for the purpose of increasing the probability of success of each project.

In [13], 100,000 open source software projects from GitHub were examined to study the popularity, interoperability and impact of various languages by measuring different metrics such as the number of lines of code, the size of development teams, the number of issues, etc.

In distributed and dynamic software project environments, in addition to source code, traces of the developers' activities (a.k.a. contribution logs) are important for the development process even though they have no direct effects on the software product itself. By combining the traditional contribution logs and the data mined from software repositories, an accurate model for measuring the developers' contributions can be built [14]. The model creates clusters of similar projects to extract weights that are then assigned to the actions that a developer performed in order to extract a combined measurement of the developer's contribution. This is particular relevant to our study, since we are considering the contributors for a given project inferred from the commits, and the total number of developers in each project.

## III. Research Method

The research methodology used in analyzing SourceForge Projects and their communities is divided into two phases:

1) **Data collection:** this phase involves collecting information of 252,008 projects.
2) **Statistical analysis:** this phase involves analyzing the data in order to measure the effect of committers, developers, and top programming languages on the success of the project..

To perform our empirical study, we collected data on 252,008 open source projects, written for numerous purposes by collaborative teams using a wide range of languages. A manual analysis of the projects on SourceForge was performed, exploring the developers, audiences, and the different programming languages used in each project. It has been found that there is a clear diversity in the developed projects that are hosted by SourceForge, and that it provides the needed requirements of this particular study.

The data that was used in the analysis was collected from the repositories of open source projects hosted on SourceForge. The tool that was used to extract the datasets is Boa. The first step was to collect datasets about the most popular programming languages, and about the number of commits and committers in each project. Data related to file changes, users, project audiences, topics, and repositories per category of use was collected. The resulting data contains records for 252,008 projects, 494,158 code repositories, 15,063073 revisions, and 147,074,540 files.

All the quantitative analyses were performed using the statistical software package RStudio [15]. In addition, we used Gephi [16], which is one of the most popular tools for graph visualization, in order to better understand data and to quickly test alternative visual approaches that best display the programming languages popularity.

We focused on the correlations in order to determine whether and how strongly pairs of each of the above variables are related to each other. We suspected that if a project is written by five or more programming languages or has many team members then the success rate should be relatively high. Hence, in order to verify our hypotheses, the Pearson's coefficient of correlation was computed to identify any strong positive correlation(s). More formally, the formula is defined in Equation 1:

$$r = \frac{\sum_{i=1}^{n}((x_i - \bar{x})(y_i - \bar{y}))}{\sqrt{\sum_{i=1}^{n}((x_i - \bar{x})^2(y_i - \bar{y})^2)}} \quad (1)$$

The equation demonstrates the ratio of the covariation between $x$ and $y$ to the total variation in both $x$ and $y$, where $\bar{x}$ and $\bar{y}$ are their respective mean values.

## IV. Results

Due to the space limitations, in this section we present the partial results of our study. The popularity was measured by considering the total number of committers and developers. It is evident that the top ten programming languages are more popular in terms of the number of committers and developers. Moreover, less people tend to view projects written in the lower-ranked programming languages. Higher number of developers and committers was an indicator of the project's success. Furthermore, the higher number of developers and committers in projects indicated greater popularity and success. Finally, the projects that are written in the top ten programming languages have a greater potential for success.

### *A. H1: On Multi-Language Projects and Success*

To test our first hypothesis, we examined the percentage of language interoperability by calculating the total number of projects that are written in 5 or more top ranked programming languages. The relationship between project interoperability and the size of the development team is measured based on the Pearson's correlation coefficient.

Developers implement various functionalities of a single project while considering the different programming languages. Let us first investigate the projects written in more than one programing language. It was assumed that if a project is written in more than one programming language, then the programming languages used in a single project are classified as interoperable.

Developers that use JavaScript, Perl, and Python are more willing to adopt different programming languages in their projects, with the percentage of such developers reaching 32.38%, 30.12%, and 28.26%, respectively. In contrast, the programming language with the least percentage of developers using multiple programming languages is PHP, with a percentage of just 15.84%. It has been noticed throughout the years that some programming languages persist, while others fail. By identifying the factors that influence this phenomenon, developers can make informed decisions when determining whether and when to invest the time and effort necessary to learn a new programming language. To date, the language adoption process has not been quantitatively studied at a large scale [17].

Table I shows programming language interoperability of using five or more programming language in a single project. The programming language that has the highest percentage in terms of interoperability is Unix Shell with a percentage of 18.66%, followed by Perl (18.18%), Delphi/Kylix (15.90%), and python (10.45%). PHP, JavaScript, C, and C# also appear in respectively 10.15%, 8.11%, 4.91%, and 4.85% of the multi language projects. C++ and java appear to be the least programming languages used in projects written in five or more programming languages.

TABLE I
Language Interoperability. Numbers and percentages of projects written in multiple programming languages, that include different programming languages.

| Programming Language | Multi-Language Projects | Percentage of Multi-Language Projects |
|---|---|---|
| Java | 65 | 1.74% |
| C++ | 99 | 3.30% |
| PHP | 59 | 10.15% |
| C | 88 | 4.91% |
| Python | 65 | 10.45% |
| C# | 48 | 4.85% |
| JavaScript | 47 | 8.11% |
| Perl | 58 | 18.18% |
| Unix Shell | 28 | 18.66% |
| Delphi/Kylix | 14 | 15.90% |

The heat-map correlation matrix in Figure 1 illustrates the language interoperability, taking into consideration the projects that are written in 5 or more top ranked programming languages. The highest relationship is found between C++ and Java, with a total of 70 projects in common. C++ also seems to work well with C, as there are 68 projects written in both languages along with other programming languages. C also work well with Java with a total number of 64 projects in common along with other programming languages. The lowest relationship is found between Perl and Delphi/Kylix with only 3 projects in common.

TABLE II
Correlation between programming languages and developers in open source projects.

| Programming Language | $r$ | $t$ | t statistics |
|---|---|---|---|
| Java | 0.16 | 29.488 | -52.443 |
| C++ | 0.18 | 13.326 | -21.605 |
| PHP | 0.22 | 8.725 | -12.147 |
| C | 0.15 | 9.3345 | -14.505 |
| Python | 0.14 | 5.4098 | -11.873 |
| C# | 0.14 | 6.4307 | -10.897 |
| JavaScript | 0.19 | 7.6198 | -11.513 |
| Perl | 0.11 | 3.1919 | -6.9357 |
| Unix Shell | 0.36 | 7.3377 | -7.2427 |
| Delphi/Kylix | 0.26 | 3.5508 | -4.7649 |
| Overall Correlation | 0.17 | 30.228 | -50.301 |

The relationship between the total number of programming languages and the total number of developers in each project is investigated. A positive correlation would indicate that increasing the number of programming languages in a project implies the need for a larger team sizes of developers.

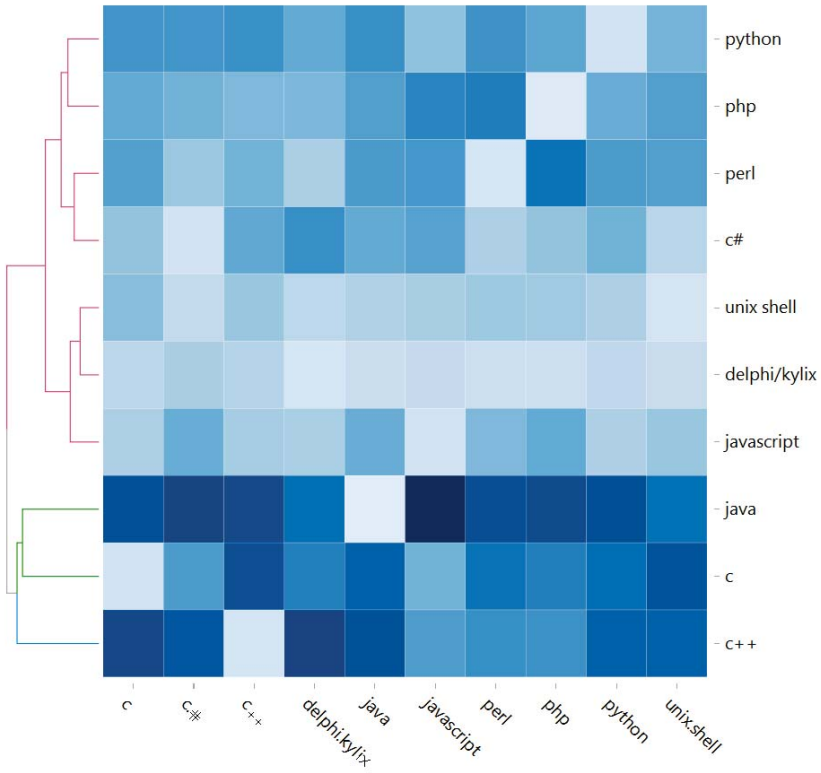


Fig. 1. Heat-map of language interoperability.

Table II demonstrates that there is a positive correlation which implies that increasing the number of programming languages requires a larger team sizes of developers. A moderate correlation is found with projects written in Unix Shell, Delphi/Kylix, and PHP.

For the data in Table II, Pearson's correlation coefficient between top programming languages and developers per projects is demonstrated. Let $\rho$ be the true correlation between programming language interoperability and developers. The null hypothesis is that there is no relationship between these two variables and the research hypothesis is that the two variables are positively related. These hypotheses are:

H0 : $\rho = 0$

H1 : $\rho \neq 0$

The test statistics is $r$ and the standardized t statistic is:

$$t = r\sqrt{\frac{n-2}{1-r^2}} \tag{2}$$

There are $n-2$ degrees of freedom. Choosing the $0.95$ confidence level for a two tailed test, the region of the correlation coefficient and the corresponding $t$ value is specified for each programming language. RStudio was used to conduct the t-test. While the correlation coefficient is not close to $1$, it is evident that it is not equal to $0$ which is enough to reject the hypothesis of no relationship.

### *B. H2: On Top Programming Languages and Success*

To test the second hypothesis, we measured the percentage of committers in projects that are written in top ranked programming languages. The relationship between projects that are written in top ranked programming languages and the total number of committers is measured using the Pearson's correlation coefficient.

Table III shows the correlation between the programming languages and committers as well as the $t$, p-value and t-test result which implies that increasing the number of programming languages requires a larger team of developers. A moderate correlation is found with projects written in Unix Shell, Delphi/Kylix, and C++. A low correlation is found with projects written in Perl, JavaScript, Python, and PHP.

For the data in Table III, Pearson's correlation coefficient between top ranked programming languages and committers per projects are computed in a similar way as in the above Section IV-B. Again, while the correlation coefficient is not close to $1$, it is evident that it is not equal to $0$ which is enough to reject the hypothesis of no relationship.

Figure 2 shows a scatter plot matrix that show how programming language interoperability is correlated with the total number of committers per projects. The green line interpret the relation, and it is evident that projects written in top ranked programming languages result in a higher number of committers. This is considered as an evidence that if a project is written using top ranked programming languages, then the probability of project success is higher, and the evaluated metric which is the total number of committers does have a positive affect on the overall project success rate.

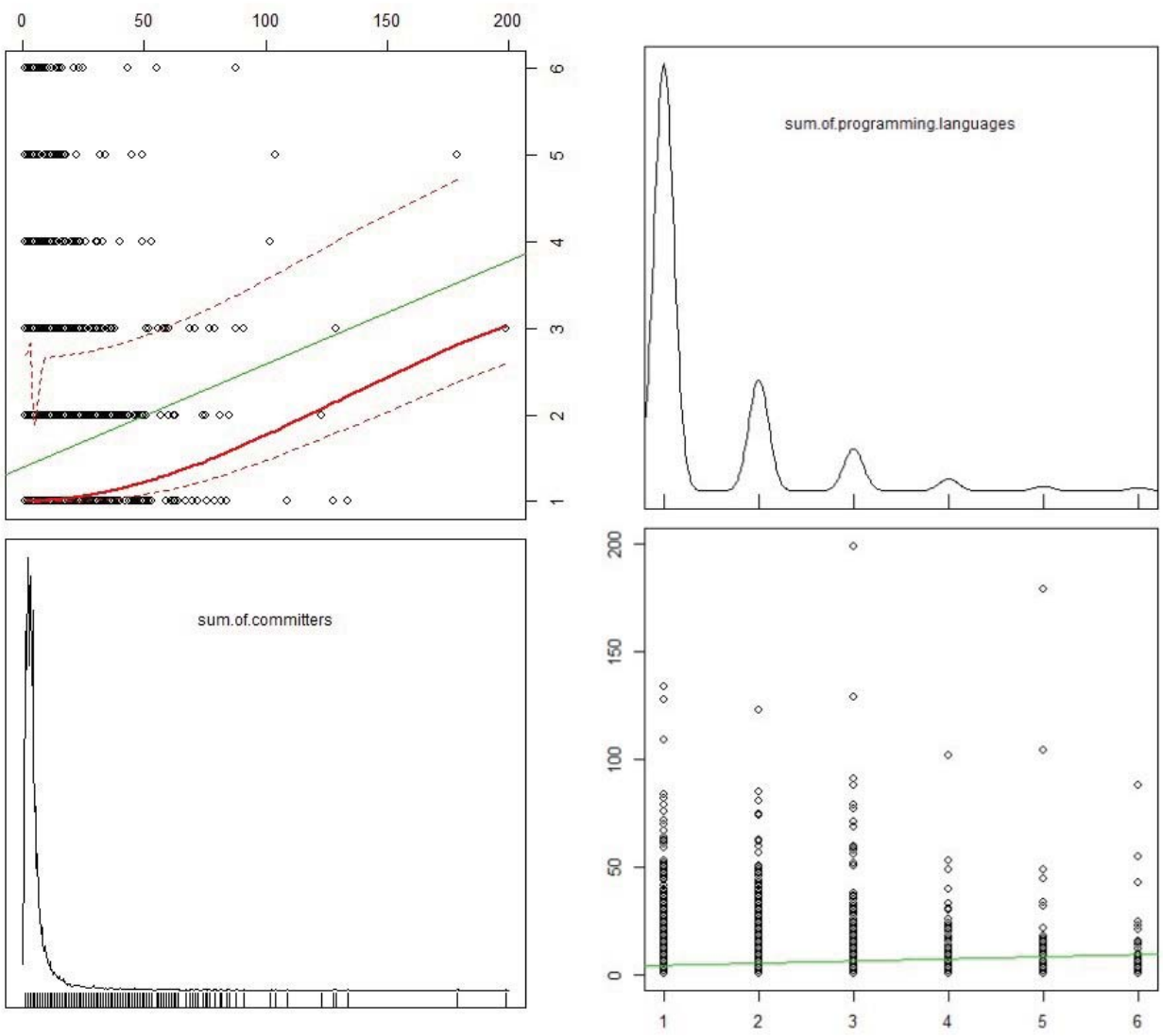


Fig. 2. Correlation between programming language interoperability and committers.

## V. Evaluation

The empirical findings are based on open source projects found on SourceForge which lead to results that may not generalize to the universe of open source projects produced by developers. The study focused on open source projects written in top ranked programming languages based on the obtained data-sets from the year 2013. The findings of the study remain relevant as Microsoft, and Apache developed several successful projects on SourceForge since then.

This study is based on 252,008 open source projects, which is a sizable sample, yet it does not equate the millions of software programs whose code can be retrieved from the World Wide Web. SourceForge alone contains over 430,000 open source projects. It is to be noted however that the sample data-set is sufficiently representative of the universe of SourceForge projects. Therefore, we believe our results generalize to other open source projects.

To ensure content validity, projects that are written in top programming languages and are written in 5 or more programming languages were selected. The percentage of developers that are in multi-language projects were analyzed. Projects that are written using a single programming language and had one developer were discarded from this study. It is difficult to conclude that the proposed hypotheses hold for projects that are written in other combinations of lower- and higher- level programming languages.

Different visualization techniques were interpreted to demonstrate the information using a variety of methods. The external validity was ensured by considering a large sample of data in order to generalize the study result to other studies of open source projects, which measure the success rate using metrics such the development team, and the total number of committers. The experiment of t-test was conducted to ensure correlation significance level.

TABLE III
CORRELATION BETWEEN PROGRAMMING LANGUAGES AND COMMITTERS.

| **Programming Language** | $r$ | $t$ | **t statistics** |
|---|---|---|---|
| Java | 0.12 | 22.646 | -72.199 |
| C++ | 0.17 | 12.534 | -26.324 |
| PHP | 0.08 | 3.1538 | -16.579 |
| C | 0.12 | 7.198 | -19.252 |
| Python | 0.08 | 3.1102 | -15.631 |
| C# | 0.13 | 5.9351 | -13.722 |
| JavaScript | 0.04 | 1.8102 | -16.054 |
| Perl | 0.03 | 0.92751 | -10.369 |
| Unix Shell | 0.28 | 5.7351 | -6.3342 |
| Delphi/Kylix | 0.19 | 2.5633 | -5.9096 |
| Overall Correlation | 0.13 | 23.143 | -69.678 |

## VI. CONCLUSIONS

In this paper, we studied a dataset of 252,008 open-source projects. Our study endorses different assumptions on literature regarding programming language popularity. The findings demonstrated that some programming languages are more popular than others. In particular, earlier programming languages such as C are common with a large code base. Moreover, programming languages such as JavaScript and Ruby are pervasive in the area of web development. The success of some programming languages is linked to the success of certain products. For example, Apple uses the Objective C programming language, which is gaining much attention nowadays. Several programing languages are created but after a while they are no longer used. However, there exists some programming languages that are preferable by a huge number of users, and these types of languages are focused on and improved upon to avoid fading away as is the case with many programming languages. Many studies focus on the methods in which programming languages should be created. Nonetheless, less attention has been given to quantifying the popularity of these programming languages, and understanding why these languages happen to be the most prevalent.

To evaluate our first hypothesis, an approach was presented based on the percentage of developers and the percentage of committers. In terms of developers, what was focused on was the developers that adopt 5 or more top ranked programming languages in the projects that were studied. The percentage of language interoperability is measured taking into consideration the number of projects that are written in 5 or more top-ranked programming languages. The relationship between project interoperability and the number of the development team is measured by applying Pearson's correlation coefficient. As for our second hypothesis, the percentage of committers in projects that are written in top ranked programming languages was measured. The relationship between project that are written in top ranked programming languages and the total number of committers was measured by applying Pearson's correlation coefficient.

Our study has implications for future research on programming languages. We demonstrated that using various programming languages requires additional team members. However, there is a notable increase in the functionality of the developed project. Also, there is a positive correlation between the developer team size, and the total number of committers.

## REFERENCES


[1] A. Stefik and S. Siebert, "An empirical investigation into programming language syntax," *ACM Transactions on Computing Education*, vol. 13, no. 4, 2013, article no. 19.

[2] 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," *Research Policy*, vol. 32, no. 7, pp. 1159–1177, 2003.

[3] D. Riehle, "Controlling and steering open source projects," *Computer*, vol. 7, no. 44, pp. 93–96, 2011.

[4] D. German and A. Mockus, "Automating the measurement of open source projects," in *Proceedings of the 3rd Workshop on Open Source Software Engineering*, 2003, pp. 63–67.

[5] A. Ankolekar, J. D. Herbsleb, and K. Sycara, "Addressing challenges to open source collaboration with the semantic web," in *Proceedings of 3rd Workshop on Open Source Software Engineering, the 25th International Conference on Software Engineering (ICSE)*. IEEE, 2003, pp. 9–14.

[6] R. Dyer, H. A. Nguyen, H. Rajan, and T. N. Nguyen, "Boa: A language and infrastructure for analyzing ultra-large-scale software repositories," in *Proceedings of the 2013 International Conference on Software Engineering (ICSE)*. IEEE, 2013, pp. 422–431.

[7] H. Siy, P. Chundi, and M. Subramaniam, "Summarizing developer work history using time series segmentation: Challenge report," in *Proceedings of the 2008 International Working Conference on Mining Software Repositories (MSR)*. ACM, 2008, pp. 137–140.

[8] M. Burch, S. Diehl, and P. Weibgerber, "Visual data mining in software archives," in *Proceedings of the 2005 ACM Symposium on Software Visualization (SOFTVIS)*. ACM, 2005, pp. 37–46.

[9] D. P. Delorey, C. D. Knutson, and S. Chun, "Do programming languages affect productivity? a case study using data from open source projects," in *Proceedings of the 2007 First International Workshop on Emerging Trends in FLOSS Research and Development (FLOSS)*. IEEE, 2007, pp. 1–5.

[10] D. G. Feitelson, G. Z. Heller, and S. R. Schach, "An empirically-based criterion for determining the success of an open-source project," in *Proceedings of the 2006 Australian Software Engineering Conference (ASWEC)*. IEEE, 2006, pp. 1–6.

[11] M. Grottke, L. M. Karg, and A. Beckhaus, "Team factors and failure processing efficiency: An exploratory study of closed and open source software development," in *Proceedings of the 2010 IEEE 34th Annual Computer Software and Applications Conference (COMPSAC)*. IEEE, 2010, pp. 188–197.

[12] A. W. R. Emanuel, R. Wardoyo, J. E. Istiyanto, and K. Mustofa, "Success factors of OSS projects from SourceForge using datamining association rule," in *Proceedings of the 2010 International Conference on Distributed Framework and Applications (DFmA)*. IEEE, 2010, pp. 1–8.

[13] T. F. Bissyande, F. Thung, D. Lo, L. Jiang, and L. Reveillere, "Popularity, interoperability, and impact of programming languages in 100,000 open source projects," in *Proceedings of the 2013 IEEE 37th Annual Computer Software and Applications Conference (COMPSAC)*. IEEE, 2013, pp. 303–312.

[14] G. Gousios, E. Kalliamvakou, and D. Spinellis, "Measuring developer contribution from software repository data," in *Proceedings of the 2008 International Working Conference on Mining Software Repositories (MSR)*. ACM, 2008, pp. 129–132.

[15] M. P. J. Van der Loo, *Learning RStudio for R Statistical Computing*. Packt Publishing Ltd, 2012.

[16] K. Cherven, *Network Graph Analysis and Visualization with Gephi*. Packt Publishing Ltd, 2013.

[17] L. A. Meyerovich and A. S. Rabkin, "Empirical analysis of programming language adoption," *ACM SIGPLAN Notices*, vol. 48, no. 10, pp. 1–18, 2013.