


Article

Healthy or Not: A Way to Predict Ecosystem Health in GitHub

Zhifang Liao ¹, Mengjie Yi ¹, Yan Wang ¹, Shengzong Liu ^{2,*}, Hui Liu ³, Yan Zhang ⁴ and Yun Zhou ⁵

¹ School of Software, Central South University, Changsha 410075, China; zfliao@csu.edu.cn (Z.L.); mengjie_yi@csu.edu.cn (M.Y.); wang_yan@csu.edu.cn (Y.W.)

² Department of Information Management, Hunan University of Finance and Economics, Changsha 410075, China

³ Department of Computer Science, Missouri State University, Springfield, MO 65897, USA; huiliu@missouristate.edu

⁴ Department of Computing, School of Computing, Engineering and Built Environment, Glasgow Caledonian University, Glasgow G4 0BA, UK; yan.zhang@gcu.ac.uk

⁵ School of Software, Central South University, Changsha 410075, China; zhouyun@hufu.edu.cn

* Correspondence: lshz179@163.com; Tel.: 86-731-82655877

Received: 23 December 2018; Accepted: 22 January 2019; Published: 28 January 2019



Abstract: With the development of open source community, through the interaction of developers, the collaborative development of software, and the sharing of software tools, the formation of open source software ecosystem has matured. Natural ecosystems provide ecological services on which human beings depend. Maintaining a healthy natural ecosystem is a necessity for the sustainable development of mankind. Similarly, maintaining a healthy ecosystem of open source software is also a prerequisite for the sustainable development of open source communities, such as GitHub. This paper takes GitHub as an example to analyze the health condition of open source ecosystem and, also, it is a research area in Symmetry. Firstly, the paper presents the healthy definition of GitHub open source ecosystem health and, then, according to the main components of natural ecosystem health, the paper proposes the health indicators and health indicators evaluation method. Based on the above, the GitHub ecosystem health prediction method is proposed. By analyzing the projects and data collected in GitHub, it is found that, using the proposed evaluation indicators and method, we can analyze the healthy development trend of the GitHub ecosystem and contribute to the stability of ecosystem development.

Keywords: open source software; GitHub; Symmetry; ecosystem health; evaluation method

1. Introduction

Software development has gradually evolved into an open software development form from the previously closed software production line with company organization, and the prosperity of the open source community expedites the process. The innovation and production are no longer an assignment of single organization. Distributed development method and cooperated innovation have steadily become the mainstream [1]. Faced with this situation, researchers treat software production as a new concept, and this way of production is referred to as the software ecosystem. There are different definitions of the software ecosystem, with the most accepted one being “a software ecosystem is a system within which the traditional walls between development entities have been broken down allowing collaboration and interoperability between parties” [2]. The main features of the software ecosystem are openness, diversity, regulation, and sustainability [3].

GitHub, an open source software ecosystem, is a hosted platform for Git's distributed version management control system for open source and proprietary software projects, and has now become the preferred method of managing software development and discovering existing code. Taking GitHub as an example, the study focuses on the health of open source software ecosystem by analyzing the activities and behaviors of developers and users, in order to provide a theoretical basis for the comprehensive understanding, evaluation, and management of the ecosystem.

This paper is organized as follows: Section 2 is related work, which introduces the health research of ecosystem and software ecosystem; Section 3 presents the GitHub ecosystem health indicators and prediction model; Section 4 is the experiments design and result analysis; Section 5 is the analysis of the validation of predication model; and the final chapter draws conclusions and presents possible future research orientations.

2. Related Work

Messerschmit et al. [4] proposed the concept of software ecosystem for the first time. Subsequently, a large number of researchers were attracted to study various fields of software ecology. Many researchers give their own definitions to the ecological system. Manikas et al. [5] defined software ecosystem as "many software solutions and services generated on the interaction of a series of participants on a generic technology platform". Bosch et al. [6,7] put forward the trend of software ecosystem, and the key concepts and influences that adapt to the software ecosystem. He divided the research challenges of software ecosystem into three levels: software ecosystem level, software supply network layer, and software provider. Plakidas et al. [8] inferred the structure and evolution of R open source ecosystem by analyzing the distribution of packages and the dependence of packages within the project through statistical calculation, which improved the understanding of the software ecosystem. Santos et al. [9] applied the concept of software reuse to the software ecosystem, and created and validated the knowledge system of the software ecosystem based on concept maps, which enriches diversity research.

Matragkas et al. [10] mapped the ecosystem to an open source software system and studied the problem of ecosystem diversity. Franco-Bedoya et al. [11] argued that software ecosystem has become an example of understanding the dynamics and heterogeneity of collaborative software development. Software ecosystem seems to be an effective tool for analyzing open source software systems, so further investigation and study of the open source community are needed to truly understand the open source software ecosystem. Jin et al. [12] discussed the research progress of open source software and the influence of open source software on technology, industry, and society, and prospected the evolution trend of the open source software ecosystem. Liao et al. [13] analyze the internal or external characteristics of free open source projects in GitHub to find out which characteristics can be used to forecast the life-span length of projects and propose a prediction model to estimate the project life-span in open source software ecosystems.

Besides, researchers have also gained great interest in open source projects. Gousios et al. [14] summarized the behaviors of participants involved in open source software projects, and proposed a method of calculating participants' contribution to multiple behaviors. Qin et al. [15] provided an effort estimation method for open source projects in GitHub. This method is especially suitable for new projects that lack training data. Capiluppi et al. [16] illustrated the characteristics of open source projects in terms of project size, programming language, number and role of developers, and evolution of open source projects. This analysis brings useful insights into the debate on the open source movement.

With regard to the health of software ecosystems, Manikas et al. [17] defined ecosystem health as the ability to maintain variables and productivity as time goes on, and the characteristics of health are dynamic, active, long-lived, and growing. Jansen [18] proposed the software ecosystem health measurement method PRN (productivity, robustness, niche creation). Wang et al. [19] probed into the affection of the ecosystem health in terms of vigor, organizational structure, resilience, and

sustainability in natural ecosystems. Liao et al. [20] measured the sustainability of open source software ecosystem from the aspects of openness, stability, activity, and extensibility, and applied the assessment method of open source software ecosystem sustainability from the target level, criterion level, and measurement level to stack overflow.

At present, the research of software ecosystem mainly focuses on system structure, participants, and ecosystem characteristics. The research on ecosystem health is mainly based on its theoretical research, and there is less quantitative analysis of the health indicators. Therefore, it is worthwhile to get into the questions of how to analyze the GitHub ecosystem health quantitatively, and how to predict the healthiness of the ecosystem.

3. GitHub Ecosystem Health Prediction Method

In this section, the working process of GitHub ecosystem and the various behaviors of users in the GitHub ecosystem will be introduced in Section 3.1, and then followed by the Health Prediction Model of GitHub Ecosystem (HPGE model). Figure 1 is an overview of the HPGE model.

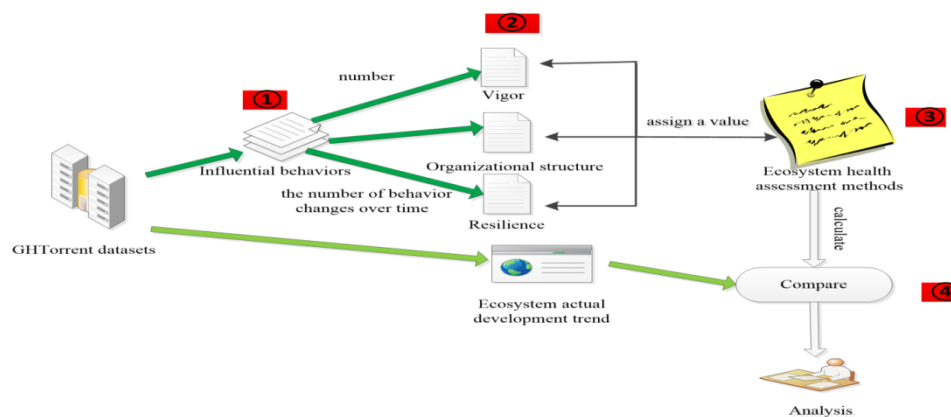


Figure 1. Overview of the proposed method Health Prediction Model of GitHub Ecosystem (HPGE). HPGE includes 4 steps: (1) Choose the factors that affect health in the GitHub ecosystem; (2) Measure the health indicator of vigor, organizational structure, and resilience in the ecosystem; (3) Calculate ecosystem healthiness based on the proposed ecosystem health prediction formula; (4) Compare the actual health variation of the project with the prediction of ecosystem health.

3.1. The Workflow of GitHub Ecosystem

In the GitHub ecosystem, the main working process of a project is as follows: First, the owner creates the open source code repository, the contributor forks the source code repository and makes the commit to the open source program and modifies the submitted pull request. Then, the reviewer evaluates the submitted pull request and submitted comment. Finally, the integrator evaluates the comment submitted by reviewer to determine whether he will merge the commit comment submitted by the contributor. During the whole process, other members can pay attention to the projects and developers. And the Figure 2 will be beneficial to improving the understanding of the workflow of GitHub ecosystem.

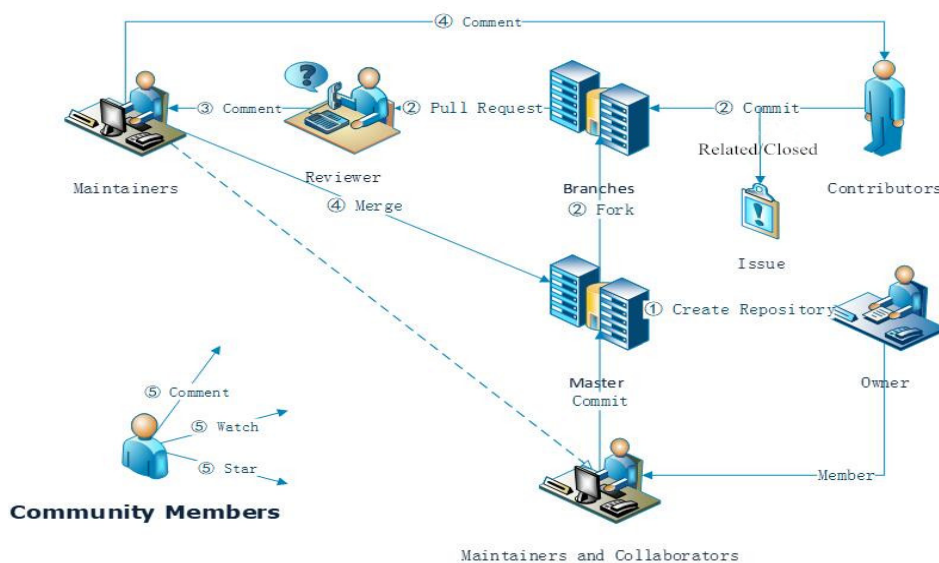


Figure 2. The workflow of GitHub ecosystem.

Owners and collaborators are the core developers of the project, and the number of forks and watchers represent the attractiveness and activity of project. Pull_request, pull_request_comment are the main behaviors in the PR mechanism. Issue is a way to put forward problems and solve problems throughout the project development. The research team selected the most representative behaviors in GitHub to analyze the GitHub ecosystem, which are fork, watcher, issues, pull_request and pull_request_comment, which are selected as the parameters of ecosystem health assessment.

3.2. The Health Indicators of GitHub Ecosystem

There is no clear standard about the definition of ecosystem health, one widely accepted definition is that ecosystem health is stable and sustainable [18]. That is to say, the ecosystem health should be able to maintain its organizational structure and have long-term stability, good self-adjustment abilities, and a certain capacity for external pressure. Consequently, we present a definition of health based on the GitHub ecosystem.

Definition 1. *The health of the GitHub ecosystem generally refers to the behavior of those participants (developers, users, etc.) and the resulting behavior that always results in a trend that is consistent with the project's survival and development, providing continuous service in the event of disruption, and also meeting people's needs for project capabilities.*

The indicators for evaluating ecosystem health in natural ecosystems mainly include vigor, organizational structure, and resilience. Among them, vigor is its activity, metabolism, and primary productivity, the organizational structure refers to the diversity of ecosystem composition, while resilience is the capacity of system to overcome pressure and rebound.

Analogous to natural ecosystems, software ecosystems should have these indicators, such as vigor, the organizational structure, and resilience. Vigor represents the flowing degree of system information. The organizational structure represents the contribution of different developers. Resilience represents the changing trend of the number of participants' behaviors. Therefore, definition 2 of software ecosystem health indicators is given as below.

Definition 2. *The indicators of ecosystem health in the GitHub ecosystem include vigor, organizational structure, and resilience. Vigor is the primary productivity, represented by the number of user behaviors in the project. The organizational structure of ecosystem health is represented by the contribution of different behaviors of the developers. Resilience is the ability of ecosystem to maintain its structure. It is the ability of*

the system to overcome pressure when it is disturbed (such as the developer’s departure), and mainly focuses on the recovery rate after interference. Figure 3 is the mapping of health indicators of the GitHub ecosystem and natural ecosystem.

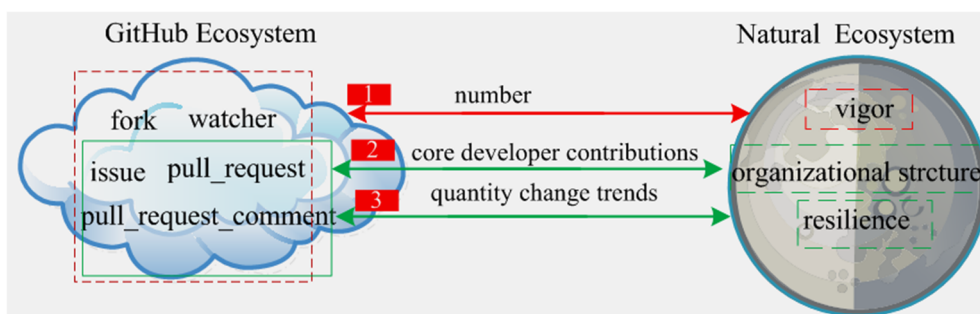


Figure 3. Mapping of health indicators of the GitHub ecosystem and natural ecosystem. It includes the following sections: (1) Use the number of fork, issue, watcher, pull_request, and pull_request_comment (elements) in the GitHub ecosystem to map to the vigor of the natural ecosystem; (2) The contribution of the developer in the issue, pull_request, pull_request_comment to represent the natural ecosystem organizational structure; (3) The changing trends in the number of influencing factors (issue, pull_request, pull_request_comment) in the GitHub ecosystem to map the resilience of the natural ecosystem.

3.3. HPGE

Considering the health indicators mapping from natural ecosystem, as well as the user behaviors in GitHub ecosystem, we present the GitHub ecosystem health prediction model (HPGE) as in Formula (1).

$$HPGE = \alpha C + \beta O + \gamma R, \tag{1}$$

where α, β, γ are coefficients; C is the vitality of the system, or number of user behaviors in the project; O is system organizational structure index, and represented by the contribution of different behavior of developers; and R is a recovery force index, and it presents the relative degree of system recovery force.

3.4. The Measurement of Indicators in the Ecosystem Health Prediction Method

3.4.1. Measurement of Vigor

Vigor is mainly represented by the number of users’ behaviors. The larger the number of user behaviors, the better the project’s vigor, but the question remains as to how many user behaviors can explain whether the vigor is good? Therefore, the measurement of vigor is mainly reflected in the threshold analysis of vigor. We count the number of the user behaviors in projects, through the statistical results, and the distribution characteristics of each behavior and the change trend of the activity of the project are obtained. Since the distribution characteristics of user behavior data are mostly normal distribution in GitHub, the Gauss distribution is used to calculate the vitality threshold.

$$\mu = \frac{1}{n} \sum_{i=1}^n X_i, \tag{2}$$

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^n X_i^2 - \left(\frac{1}{n} \sum_{i=1}^n X_i \right)^2, \tag{3}$$

where μ is the mean of the random variable that follows Gaussian distribution, σ^2 is the variance of the random variable, n is the total sample size, and X_i represents the size of sample i. We compare

different behavior distribution and try to choose the best range to represent the threshold space of the vigor.

3.4.2. Measurement of Organizational Structure

In the GitHub, if the behavior contribution of core developers is larger in the code merging phase, the quality of projects can be guaranteed, and this situation is beneficial to the health of the GitHub ecosystem. In this paper, we first use the information entropy to measure the contribution of each behavior and screen out the core behaviors in the project, and then calculate the contribution of core developers in the influential behaviors. If core developers make a large contribution in the code merge phase and small contributions in other phases, it indicates that more non-core developers participate in other phases, and the situation can ensure the diversity of ecosystems and ecosystem health.

In this paper, information entropy is used to measure the contribution of each action, as in Formula (4).

$$E_j = -K \sum_{i=1}^m P_{ij} \ln(P_{ij}), \quad (4)$$

where $P_{ij} = \frac{X_{ij}}{\sum_{i=1}^m X_{ij}}$ represents the contribution of the i th project under the j behavior, X_{ij} represents the number of j behaviors in i project, E_j represents the total contribution of behavior X_j in all projects, and $K = \frac{1}{\ln(m)}$, m is the number of projects.

When the contribution degree of each project tends to be consistent, E_j tends to 1. There is little difference in the contribution of behavior in the project. We define the degree of consistency of d_j as the contribution degree of each project under the j behavior.

$$d_j = 1 - E_j, \quad (5)$$

By calculating the contribution of each behavior in the project, we can select and remove the behavior that the contribution of each project tends to be consistent (that is, the smaller value of d_j), and we choose the behaviors which have large d_j and calculate the core developers' contribution of these behaviors in the project.

3.4.3. Measurement of Resilience

Resilience is the ability of ecosystem to maintain structure and pattern. In GitHub, the strength of resilience is determined by the behavior changes in the quantities over time. If the number of a behavior maintains the balance in a period of time, or if the number is reduced, it can restore the original number or more than the original, and we can say that the GitHub ecosystem has a strong resilience.

In the measurement of contribution, we select out the behaviors that have a great impact on the ecosystem health, analyze the changes of the monthly average of the individual behaviors, and try to find the law of the resilience of the ecosystem in a period of time.

4. Experiments Design and Result Analysis

4.1. Dataset Analysis

We use GHTorrent to collect data. The experimental data we used are generated from the beginning of these projects to December 2016, and the link for the data is <https://github.com/yimengjie>. The dataset is divided into two parts: the training set and the test set. The training set, based on project data, is up to December 2014. The data as test set from 2015 to 2016 is used to verify the correctness of the Formula (1). We chose a lot of projects to do the experiments, but only chose 24 classical projects to explain our main ideas. We choose the following actions as the influence factors: fork, watcher, pull_request, and pull_request_comment, and we also calculate the number of

core developers in each project. The following work needs to be done to analyze the characteristics of ecosystem in GitHub.

- (1) First, it involves the analysis of the user behaviors of GitHub ecosystem and the influence factors of users on the ecosystem.
- (2) Then, indicators including vigor, organizational structure, and resilience of GitHub ecosystem are measured.
- (3) Finally, the prediction model for the health of GitHub ecosystem is presented by measurement of ecosystem indicators. Table 1 is a statistical analysis of collected data. We select 5 projects to be displayed. Among them, pull_request is represented by PR and pull_request_comment is represented by PRS_COMMENT.

Table 1. Data statistics of influential factors.

Project Name	# of Fork	# of Watcher	# of Issue	# of PR	# of PRS_COMMENT	# of Core Developers
bitcoin	3987	8270	6756	4331	7323	19
backbone	5434	24,171	3689	1613	1061	10
atom	3827	20,090	7541	1838	1671	122
ansible	4503	12,571	11,372	6353	429	37
caffe	2588	4764	2626	1060	1168	19

4.2. Measurement of Vigor

The following work has been done.

- (1) We count the number of fork, issue, pull_request, pull_request_comment (elements) and other behaviors in 24 projects, and calculate the values of Min, Max, Mode, and StdDev under each behavior.
- (2) The Gaussian distribution is used to derive the rules for the changes in the number of submissions in each project.
- (3) The rules obtained in (2) are used to classify the vigor of user behavior in the project and assign the value of the project's vigor size.

The steps, in detail, are described as below.

Step 1: Several behaviors are selected, such as fork, issue, pull_request, pull_request_comment. Four values (Min, Max, Mode, StdDev) are calculated to explain project vigor. Min and Max represent the interval of the number of users' actions, Mode represents the main distribution characteristics of user behavior and StdDev represents the dispersion degree of data submitted by each project. Table 2 shows only the statistical characteristics of the activity of fork, and other factors are the same as fork.

Table 2. Descriptive statistics values of fork on GitHub ecosystem.

Project Name	Min	Max	Average	Mode	StdDev
bitcoin	3	375	73.73	13	79.07
backbone	11	185	97.98	71	48.08
atom	113	691	273.36	305	145.32
ansible	3	229	109.83	184	62.79
caffe	1	302	129.4	46	95.92

Step 2: Through the statistical analysis of four kinds of behaviors, we can get the value of the number of behaviors falling in $(\mu - \sigma, \mu + \sigma)$, $(\mu - 1.96\sigma, \mu + 1.96\sigma)$ and $(\mu - 2.58\sigma, \mu + 2.58\sigma)$ that probability is 68.27%, 95% and 99% respectively in the Gauss distribution. The comparison of the values is in Table 3. When the minimum value of the interval is less than zero, the minimum value of the project is defined as Min of the project.

Table 3. Fork behavior in the range of different probabilities.

Project Name	Min	Max	68.27%	95%	99%
bitcoin	3	375	[3, 152]	[3, 229]	[3, 278]
backbone	11	185	[50, 146]	[11, 193]	[11, 223]
atom	113	691	[128, 419]	[113, 559]	[113, 648]
ansible	3	229	[47, 173]	[3, 233]	[3, 272]
caffe	1	302	[33, 225]	[1, 318]	[1, 377]

From Table 3, the probability of the number of behaviors of each influencing factor in 68.27%, 95%, and 99% are obtained. According to the distribution of probabilities, 95% and 99% probability distributions cover the whole range of the monthly fork submission number, and special information regarding the range where the monthly fork submission numbers concentrate cannot be obtained. Hence, we chose 68.27% (probability distribution) to represent the scope of the main concentration of the submissions in the project under this behavior, which helps to analyze the role of influencing factors (i.e., user behavior) in the ecosystem. Figure 4 show the changes in the monthly average submitted influence factors in the range of 68.27%.

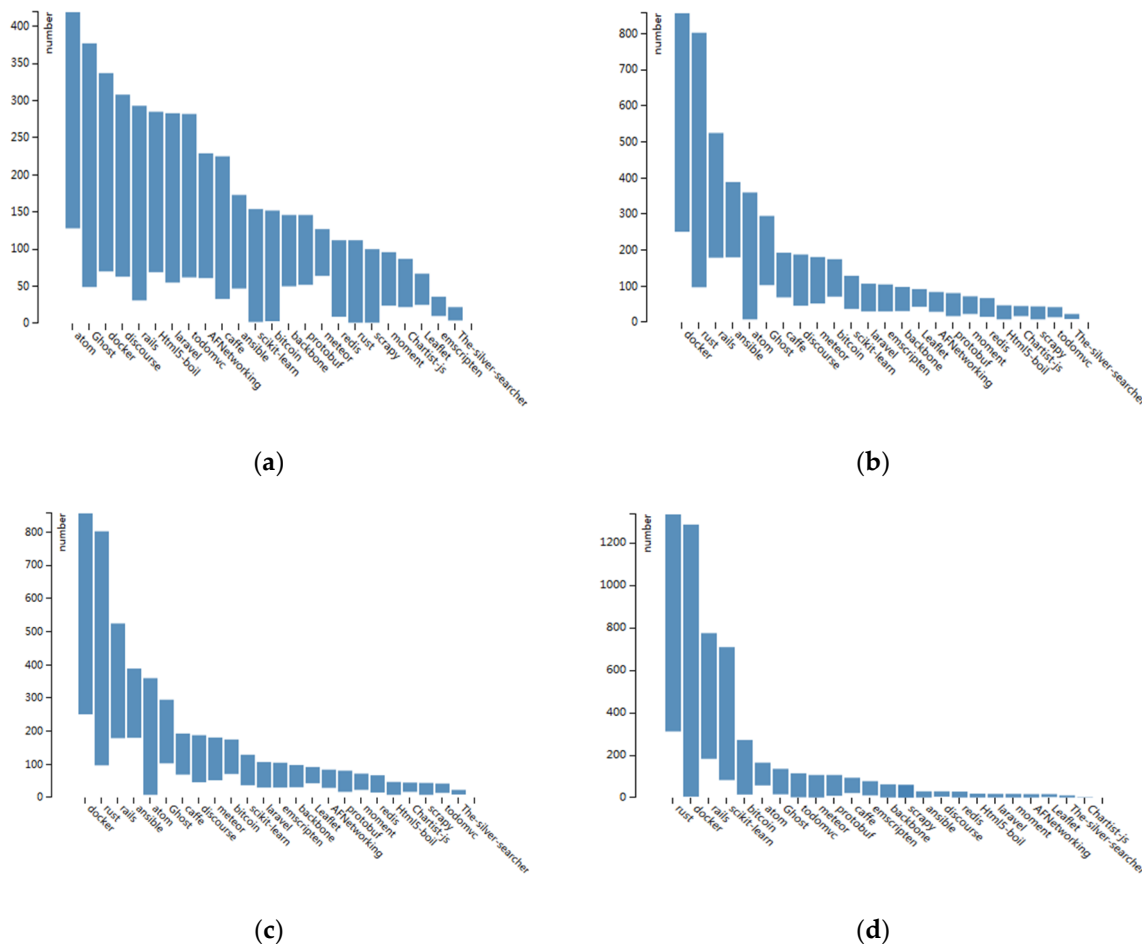


Figure 4. Monthly average changes in the number of fork, issue, pull_request, and pull_request_comment (elements) for different projects. (a): fork; (b): issue; (c): pull_request; (d): pull_request_comment.

Step 3: According to the monthly average change numbers of different behaviors, we classify the number of behaviors in a project into three levels, high/middle/low level. Figure 4d shows that in project rust, docker, rails, and scikit-learn, the pull_request_comment submits a large

amount monthly, so we classify pull_request_comment as high level and, in the project discourse, ansible, redis, Html5-boil, moment, Laravel, Leaflet, pull_request_comment is low level, and pull_request_comment in other projects are specified as mid-range. Other behaviors are classified the same as pull_request_comment. Table 4 shows the behavior distribution in 68.27% probability in a project after classification.

Table 4. Interval division of influence factors.

Influence Factors	High Interval	Middle Interval	Low Interval
fork	[62, 303]	[36, 150]	[12, 79]
issue	[136, 539]	[41, 131]	[13, 49]
Pull_request	[82, 256]	[14, 67]	[8, 28]
Pull_request_comment	[119, 876]	[13, 103]	[1, 20]

Result 1: In GitHub, we chose the number of fork, issue, pull_request, pull_request_comment (elements) to represent the vigor. From Figure 4, we can get the monthly average change numbers of different behaviors, and we tried to classify the level into three levels, four levels, and five levels, and we found these results had little differences, so we chose the minimum number to classify the levels. A project that contains a large number of activities that fall into high or middle level results in a greater vigor number. At the same time, according to the distribution of behavior, we set the vigor values of the high, medium, and low intervals to 1, 0.75, and 0.5, respectively. The values of 1, 0.75, and 0.5 were used to calculate the vigor in Formula (1), and we performed a large number of experiments and found the values of 1, 0.75, and 0.5 could better show the results.

4.3. Measurement of Organizational Structure

The following work has been done.

- (1) Calculate the value of the contribution E_j and contribution consistency d_j of fork, watcher, issue, pull_request, pull_request_comment in the 24 projects. The behaviors with smaller d_j will not be considered.
- (2) Select the behavior with larger d_j and calculate the core developer's contribution to the project under this behavior. The ste, in detail, are as follows.

Step 1: Count the number of fork, watcher, issue, pull_request and pull_request_comment (elements) of the 24 projects. Table 5 is E_j and d_j of each behavior in the projects.

Table 5. Contribution of behaviors and the contribution consistency in projects.

	Fork	Watcher	Issue	Pull_Request	Pull_Request_Comment
E_j	0.8900939	0.971034313	0.880502612	0.877495125	0.738401398
d_j	0.1099061	0.028965687	0.119497388	0.122504875	0.261598602

Result 2 for step 1: The smaller value of d_j indicates the small contribution in j behavior in each project. Table 5 shows the d_j of watcher is smallest, and this indicates the behavior of watcher has little influence on the healthy ecosystem. Similarly, the behavior of fork is not an important factor for the ecosystem health. Hence, we chose pull_request, pull_request_comment, and issue to study the core developer contributions of GitHub ecosystem.

Step 2: Calculate the core developer's contribution to the project in issue, pull_request, pull_request_comment. Formula (6) is used to represent the contribution of core developers to behavior j .

$$P_j = \frac{N_{j:\text{core}}}{N_j}, \quad (6)$$

where $N_{j:core}$ represents the number of behaviors submitted by core developers, and N_j is the mean of the number of behaviors submitted by all participants, in which the contribution of pull_request is shown in Table 6. Issue and pull_request_comment are measured in the same way as pull_request.

Table 6. The contribution of core developers to pull_request.

Project Name	# of Contributors/# of Behaviors	# of Merge/# of Behaviors	# of Consolidated Users/Total Contributor (%)	# of Merge Amount/Total Contributions (%)
bitcoin	514/4336	175/1813	34.05	41.81
Backbone	782/1611	223/572	28.52	35.44
atom	339/1848	94/787	27.73	42.59
ansible	1770/6363	680/2523	38.42	39.65
caffe	249/1066	33/126	13.25	11.82

Table 7 is the results of the contribution of core developers in issue, pull_request, and pull_request_comment.

Table 7. Contribution of core developers to influence factors.

Influence Factors	Core Developer Contributions
issue	25.8757827%
Pull_request	33.04458333%
Pull_request_comment	62.6086001%

Result 2 for step 2: From Table 7, core developers contribute 62.6086001% in pull_request_comment, while issue and pull_request contribute less than other participants, and this indicates that non-core developers contribute more in code modification and comment, which is beneficial to further software improvement and development.

4.4. Measurement of Resilience

The following work has been done.

- (1) Analysis of changes of pull_request in 24 projects.
- (2) Another 12 projects were selected to observe the change trend of the number of issue behaviors over a period of time.
- (3) Analysis of behaviors changing tendency in a project, and observation of the resilience condition in the GitHub ecosystem, the value for resilience assigned.

The steps, in detail, are described as follows.

Step 1: Because non-core developers contribute more to issue and pull_request, they can better reflect the resilience of the system. So this part chooses issue and pull_request behavior to carry out the resilience measurement experiment. The statistics of pull_request behavior in 24 projects are shown in Figure 5.

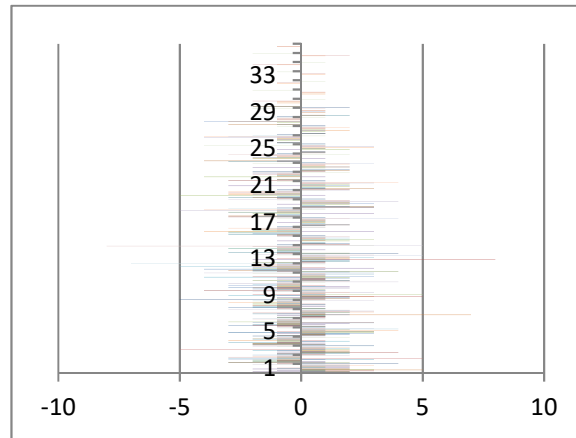


Figure 5. The changing in the number of pull_request (elements) in the project.

The horizontal axis represents the number of consecutive months of pull_request increasing or decreasing, while the vertical axis represents the month of project from publishing to interception. 0 indicates the number of pull_request (elements) at the beginning of the project, and the negative number indicates the number of months falling. For example, -2 indicate that the number has decreased continuously for 2 months, and 2 indicates that the number has increased continuously for 2 months. As we can see from Figure 5, the overall change in the number of pull_request (elements) in different projects is a fluctuation. It is generally maintained in a state of balance.

Step 2: As shown in Figure 6, 12 projects from 2010 to 2015 were selected to compare the changes in the number of issues, showing an upward trend from the initial status. This shows that the project itself has some resilience. Project atom has been selected as a case to explain the changes of issue in Figure 7.

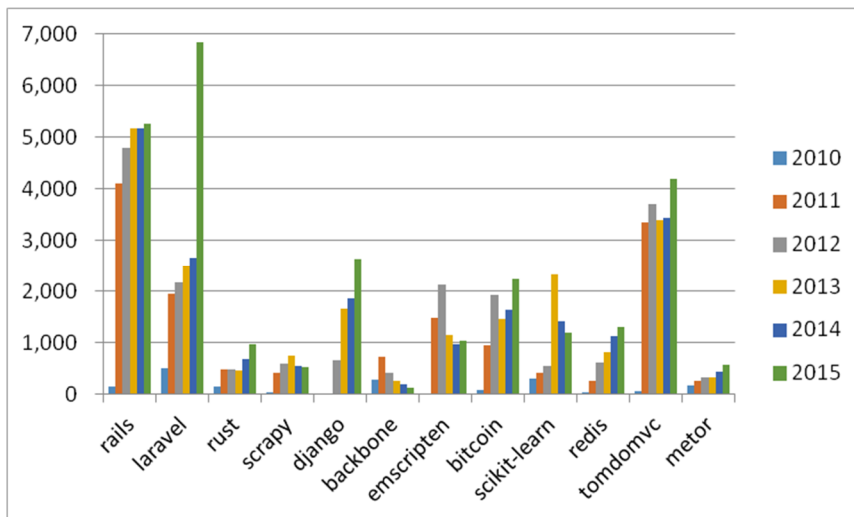


Figure 6. The changing trend of the number of issues in the project.

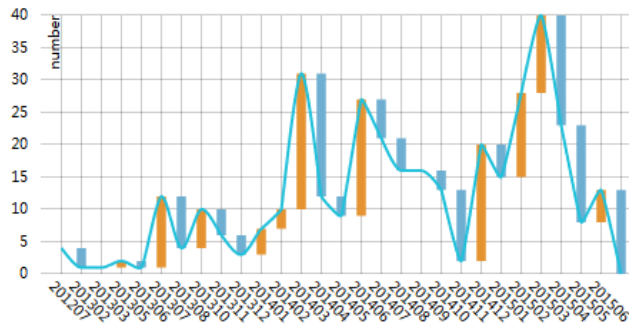


Figure 7. The atom project issue volume chart.

The orange represents an increase in data, and the blue represents a decrease in Figure 7. The height of a column represents the amount of an increase or decrease. From Figure 7, we can see that, from the project creation to June 2015, the number of increasing months is 25, the number of decreasing months is 23, and the overall trend of projects is on the rise.

Step 3: Through statistical analysis of the changes of the number of pull_request (elements) of the 24 projects, the general distribution of the three cases of resilience is available in Figure 8.

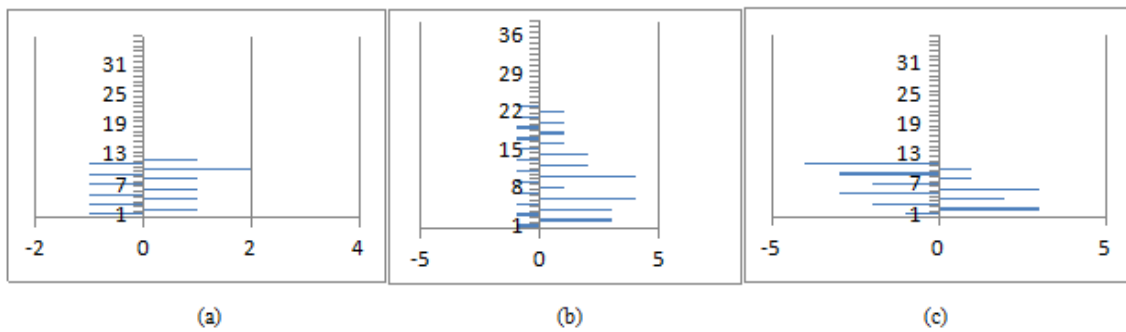


Figure 8. Three types of resilience distribution (a) Rise Months \approx Decline Months; (b) Rise Months $>$ Decline Months; (c) Rise Months $<$ Decline Months.

It states that if there is a change in the number of behaviors in a project, the number of rise months is no less than the number of decrease months, and the resilience is 1; if the number of rise months is approximately equal to the number of decrease months, the resilience is 0; if the number of rise months does not exceed the number of decrease months, the resilience is -1 .

4.5. Summary of Ecosystem Health

According to Formula (1), we know that the health of ecosystem system is composed of vigor, organizational structure, and resilience. After many experiments have been conducted, we made the following conclusions: (1) we tried to classify the level into three levels, four levels, and five levels, and we found the results had little differences, so we chose the minimum number to classify the levels. The values of 1, 0.75, and 0.5 were used to calculate the vigor in Formula (1), and we performed a large number of experiments and found the values of 1, 0.75, and 0.5 could better show the results. (2) The organizational structure is measured by the contribution of developers, and we chose the top three factors to calculate it. (3) The resilience in one ecosystem can be classified as the resilience is good, or the resilience stays in the same average as before, or the resilience cannot recover to the situation as before. Hence, we divided the experiment results into three levels. For easy calculation in Formula (1), we set resilience good as 1, and the resilience stays in the same average as 0, and the third as -1 , and we also use numbers (such as 1, 2, and 3) for calculation.

Based on the above experimental methods and results, we calculated the health indicators of vigor, organizational structure, and resilience for the new project, and then adjusted the coefficient to observe the relationship between the actual development of the project and the predicted trend.

5. Verification Analysis

According to Formula (1), we know that the health of ecosystem system is composed of vigor, organizational structure, and resilience.

Eight projects of high concern have been taken as an ecosystem. Vigor, organizational structure, and resilience values in those projects are calculated. According to Formula (1), the project’s health status is predicted and compared with the actual development trend of the project to verify the accuracy of the predictions.

Figure 9 shows, separately, measurement of the vigor, organizational structure, and resilience in eight projects.

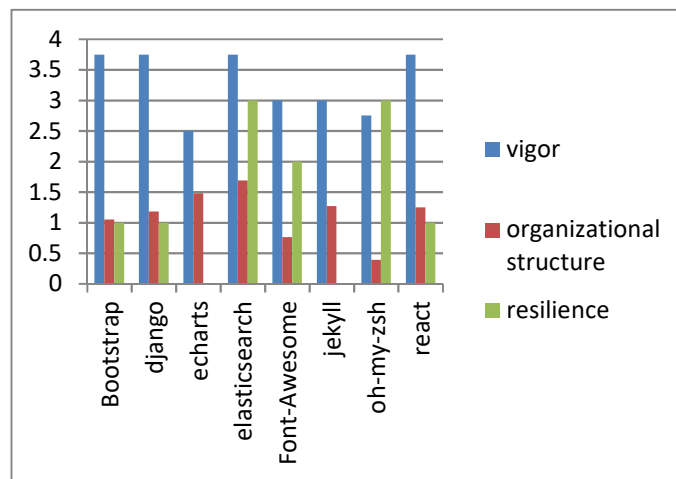


Figure 9. Measurement of the vigor, organizational structure, and resilience.

According to the prediction of Formula (1), different values of α , β , and γ have been chosen. Figure 10a shows the predicted health outcomes for $\alpha = 0.333$, $\beta = 0.333$, and $\gamma = 0.333$. Figure 10b shows the predicted health outcomes for $\alpha = 0.25$, $\beta = 0.25$, and $\gamma = 0.5$. Figure 10c shows $\alpha = 0.25$, $\beta = 0.5$, and $\gamma = 0.25$ to predict the project’s health outcomes.

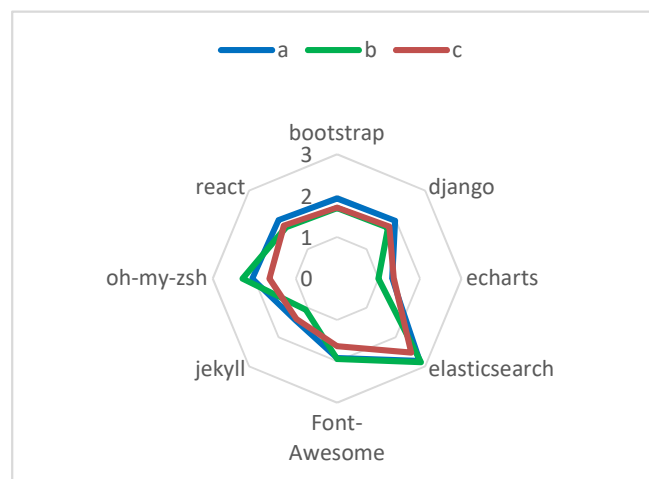


Figure 10. The health value of the project under different parameters.

In Figure 10a and b, the health orders are the same, that is, elasticsearch > oh-my-zsh > reach > django > bootstrap > Font-Awesome > jekyll > echarts. Figure 10c shows the health order is elasticsearch > reach > django > bootstrap > oh-my-zsh > Font-Awesome > jekyll > echarts.

The data of eight projects from January 2015 to June 2015 are counted to describe the actual changing trend of the project as in Figure 11.

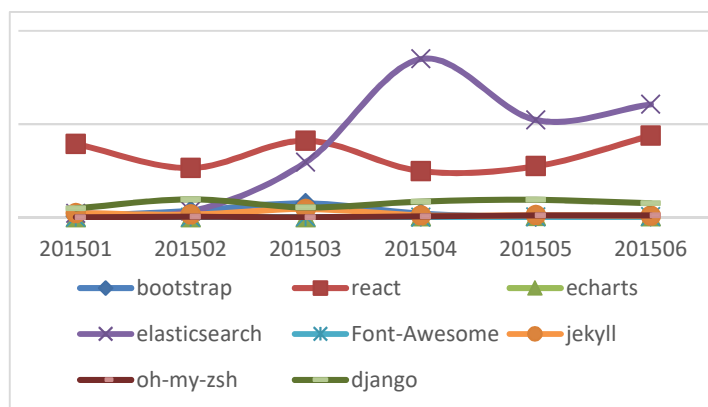


Figure 11. The actual health change chart of the project.

From Figure 11, project elasticsearch shows the best health result in eight projects in June 2015, and the next is react, followed by django, bootstrap. Due to the smaller changes in the jekyll and echarts, both are not fully presented in the actual health change charts. Compared with the health prediction method proposed in Figure 12, the health outcomes of each project are consistent with $\alpha = 0.25$, $\beta = 0.5$, $\gamma = 0.25$. This shows that the proposed prediction method is effective.

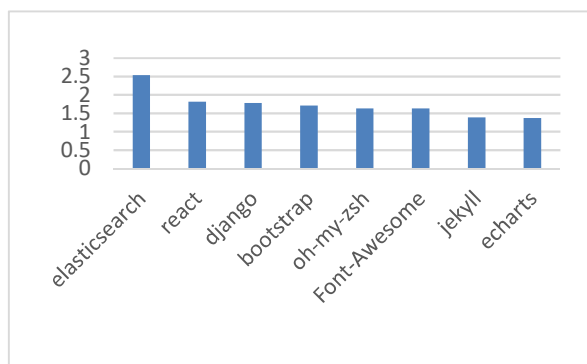


Figure 12. The sorting of project's health in $\alpha = 0.25$, $\beta = 0.5$, $\gamma = 0.25$.

6. Conclusions and Future Work

According to the existing research, there is no quantitative analysis on the health prediction of open source software ecosystem. Inspired by the natural ecosystem, this paper attempts to explore the open source software community from the perspective of an ecosystem, to form a healthy software ecosystem. We analyzed the behavior that affects the health of GitHub software ecosystem and gives the definition of GitHub ecosystem health. From the point of view of composing the vigor, organizational structure, and resilience of health indicators of natural ecosystems, it defines the indicators affecting the health of GitHub ecosystem. The health indicators were measured, and the GitHub ecosystem health prediction method was proposed. By comparing with the actual health status of the project and proposed health prediction method can predict the health of the GitHub ecosystem to a certain extent. In addition, in this paper, a limited number of relatively important factors were chosen in the selection of ecosystem health indicators.

In future work, we will use data visualization to improve the presentation of our experimental results [21] and explore the internal connecting relationships of factors. We will continue to further the analysis of influencing factors and consider the impacts of different ways of collaborating to optimize our forecasting methods. The thing we want to know is the relationship between the ecosystem health of open source software and the privacy protection of contributors [22]. However, health is only one aspect of quality. We will explore the factors affecting the quality of ecosystem services of open source software in order to maintain the sustainable development of ecosystems [23].

Author Contributions: All the authors discussed the algorithm required to complete the manuscript. Z.L. proposed the concept of this paper. M.Y. wrote the paper. Y.W. collected and processed the data for initial use and later re-use. S.L., M.Y. and Y.Z. (Yun Zhou) verified the experimental results together. Y.Z. (Yan Zhang) and H.L. revised the paper.

Funding: The works that are described in this paper are supported by NSF 61876190 and NSF 61802120, Hunan Provincial Key Laboratory of Finance & Economics Big Data Science and Technology (Hunan University of Finance and Economics) 2017TP1025 and HNNSF 2018JJ2535, The scientific research project of Hunan Provincial Education Department No.:13C095, The Fundamental Research Funds for the Central Universities of Central South University with No.2018zzts623.

Acknowledgments: We would like to thank the referees for their valuable comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mei, H.; Jin, Z.; Zhou, M.H. Open-source software ecosystems: Research and practice. *Commun. China Comput. Fed.* **2016**, *12*, 22–23.
2. Campbell, P.R.J.; Ahmed, F. A three-dimensional view of software ecosystems. In Proceedings of the Software Architecture: 4th European Conference, Copenhagen, Denmark, 23–26 August 2010.
3. Zhang, D.; Li, B.; He, P. Characteristic Study of Open-Source Community Based on Software Ecosystem. *Comput. Eng.* **2015**, *41*, 106–113.
4. Messerschmitt, D.G.; Szyperski, C. *Software Ecosystem: Understanding an Indispensable Technology and Industry*; MIT Press: Cambridge, MA, USA, 2003.
5. Manikas, K.; Hansen, K.M. Software ecosystems—A systematic literature review. *J. Syst. Softw.* **2013**, *86*, 1294–1306. [[CrossRef](#)]
6. Bosch, J.; BoschSijtsema, P. From integration to composition: On the impact of software product lines, global development and ecosystems. *J. Syst. Softw.* **2010**, *83*, 67–76. [[CrossRef](#)]
7. Bosch, J. Architecture challenges for software ecosystem. In Proceedings of the Software Architecture: 4th European Conference, Copenhagen, Denmark, 23–26 August 2010.
8. Plakidas, K.; Stevanetic, S.; Schall, D. How do software ecosystems evolve? A quantitative assessment of the R ecosystem. In Proceedings of the SPLC'16, Beijing, China, 16–23 September 2016.
9. Santos, R.P.D. ReuseSEEM: An Approach to Support the definition, modeling, and analysis of Software Ecosystems. In Proceedings of the ICSE Companion'14, Hyderabad, India, 31 May–7 June 2014.
10. Matragkas, N.; Williams, J.R.; Kolovos, D.S.; Paige, R.F. Analysing the 'Biodiversity' of Open Source Ecosystems: The GitHub Case. In Proceedings of the MSR 2014, Hyderabad, India, 31 May–1 June 2014.
11. Franco-Bedoya, O.; Ameller, D.; Costal, D.; Franch, X. Open source software ecosystems: A Systematic mapping. *Inf. Softw. Technol.* **2017**, *91*, 160–185. [[CrossRef](#)]
12. Jin, Z.; Zhou, M.H.; Zhang, Y.X. Open source software and its ecosystems: Today and Tomorrow. *Sci. Technol. Rev.* **2016**, *34*, 42–48.
13. Liao, Z.; Zhao, B.; Liu, S.; Jin, H.; He, D.; Yang, L.; Zhang, Y.; Wu, J. A Prediction Model of the Project Life-Span in Open Source Software Ecosystem. *Mobile Netw. Appl.* **2018**. [[CrossRef](#)]
14. Gousios, G.; Kalliamvakou, E.; Spinellis, D. Measuring developer contribution from software repositories data. In Proceedings of the 4th Mediterranean Conference on Information Systems, Athens, Greece, 25–27 September 2009.
15. Qi, F.; Jing, X.Y.; Zhu, X.K.; Xie, X.Y.; Xu, B.W.; Ying, S. Software effort estimation based on open source projects: Case study of GitHub. *Inf. Softw. Technol.* **2017**, *92*, 145–157. [[CrossRef](#)]

16. Capiluppi, A.; Lago, P.; Morisio, M. Characteristics of open source projects. In Proceedings of the European Conference on Software Maintenance & Engineering, Benevento, Italy, 28 March 2003.
17. Manikas, K.; Hansen, K.M. Reviewing the health of software ecosystems—A conceptual framework proposal. *CEUR Workshop Proc.* **2013**, *987*, 26–37.
18. Jansen, S. Measuring the health of open source software ecosystems: Beyond the scope of project health. *Inf. Softw. Technol.* **2014**, *56*, 1508–1519. [[CrossRef](#)]
19. Wang, C.H.; Wang, G.; Yu, Y.L. Studies on Factors Affecting Ecosystem Health. 2013. Available online: <http://www.cnki.com.cn/Article/CJFDTotal-KJZF201303003.htm> (accessed on 1 November 2018).
20. Liao, Z.; Deng, L.; Fan, X.; Zhang, Y.; Liu, H.; Qi, X.; Zhou, Y. Empirical Research on the Evaluation Model and Method of Sustainability of the Open Source Ecosystem. *Symmetry* **2018**, *10*, 747. [[CrossRef](#)]
21. Liao, Z.; Dayu, H.; Chen, Z.; Fan, X.; Zhang, Y.; Liu, S. Exploring the Characteristics of Issue-related Behaviors in GitHub Using Visualization Techniques. *IEEE Access* **2018**, *6*, 24003–24015. [[CrossRef](#)]
22. Kuang, L.; Zhu, Y.; Li, S. A Privacy Protection Model of Data Publication Based on Game Theory. *Secur. Commun. Netw.* **2018**, *2018*. [[CrossRef](#)]
23. Kuang, L.; Yu, L.; Huang, L.; Wang, Y.; Ma, P.J. A personalized QoS Prediction Approach for CPS Service Recommendation Based on Reputation and Location-Aware Collaborative. *Sensors* **2018**, *18*, 1556. [[CrossRef](#)] [[PubMed](#)]



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).