2016 30th International Conference on Advanced Information Networking and Applications Workshops

Analysis of Software Developer Activity on a Distributed Version Control System

Shu Li*, Hayato Tsukiji*, and Kosuke Takano**
Department of Information and Computer Sciences
Kanagawa Institute of Technology
Atsugi, Kanagawa, Japan
*{s1221109, s1321033}@cce.kanagawa-it.ac.jp, **takano@ic.kanagawa-it.ac.jp

***Abstract*—With the widespread use of distributed version control systems for software development, many software development projects are being shared and managed in a Web environment. This allows many developers, including project leaders and supporters, those who write important functions, those who resolve bugs and errors, and those who provide fundamental source code to relieve bottlenecks, to collaboratively progress their projects. In this paper, for the analysis of various characteristics of software developers, such as their skills and project roles, we present a method of feature extraction based on those developers' history of collaborative development using a distributed version control system. Real Git projects on GitHub were experimentally analyzed, and the experimental results demonstrate the feasibility of our proposed method.**

Keywords-GitHub; collaborative development; distributed version control system; developer; feature extraction

I. INTRODUCTION

With the widespread use of distributed version control systems for software development, many software development projects are being shared and managed in a Web environment. This allows many developers, including project leaders and supporters, those who write important functions, those who resolve bugs and errors, and those who provide fundamental source code to relieve bottlenecks, to collaboratively progress their projects.

Moreover, because GitHub publically provides the GitHub API (Application Programming Interface) [1] to users to enable the analysis of source code repositories on Git projects, many researchers are actively pursuing various analyses of online collaborative software development. Figure 1 summarizes the data that are available using the GitHub API. For example, by focusing on the "programming languages and the amounts in which they are used" in each software development project and statistically analyzing these data, we can extract the relative popularity of each programming language and rank them. In addition, several researchers are attempting to extract the emotions of software developers by analyzing their "commit messages" [2][10].

In this study, we present a method of feature extraction for analyzing the characteristics of software developers based on a history of collaborative development using a distributed version control system. Our method allows us to analyze the characteristics of software developers, such as their skills and project roles, to allow early evaluations of ongoing development projects and support human resource management efforts to place the right developers in the right development areas.

Real Git projects on GitHub were experimentally analyzed, and the experimental results demonstrate the feasibility of our proposed method.

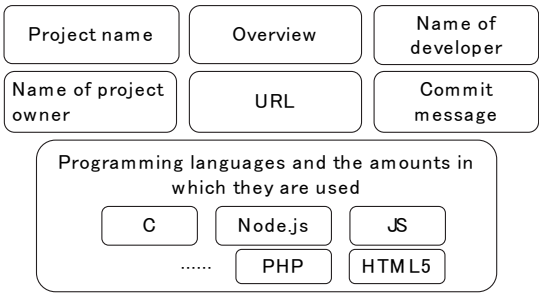

Figure 1. Data that are available using the GitHub API

II. MOTIVATING EXAMPLE

Many studies have addressed the analysis of developer behavior in online software development [6][7][8].

Onoue et al. proposed a method of developer classification based on an analysis of their activity history on GitHub, and by classifying developers participating in the active OSS (Open Source Software) projects "homebrew" and "node", they showed that the developers participating in active projects can be distinguished into different types, such as fast-commenter, fast-generalist, and slow-generalist [6][7]. Zhou et al. analyzed developer activities in OSS projects and showed that positive contributions by developers depend on the popularity of the software project [8]. Salleh et al. investigated the influence of pair programming in terms of five factors of personality: "Openness to experience", "Conscientiousness", "Extraversion", "Agreeableness", and "Neuroticism" [9]. In [10] and [11], sentiment analyses of software developers were conducted based on questions and answers from programmers on Stack Overflow [12], which

978-1-5090-2461-2/16 $31.00 © 2016 IEEE
DOI 10.1109/WAINA.2016.107

IEEE computer society

is a Q&A website focused on computer programming, and developers' commit comments on GitHub, respectively.

In this study, we focus on the MVC (Model-View-Controller) architecture for extracting the characteristics of software developers working on a Git project. "Ruby on Rails" [3], "Java Spring" [4], and "Laravel" [5] are examples of frameworks using the MVC architecture. In the MVC architecture, we can easily determine the functionality and importance of any source code file from the folder structure, filename, file extension, and so on. Our proposed method allows us to analyze each developer's characteristics in terms of "strong development areas (M/V/C)", "contribution", "initiative", "support", and "leadership" in progressing software development on a Git project (Figure 2).

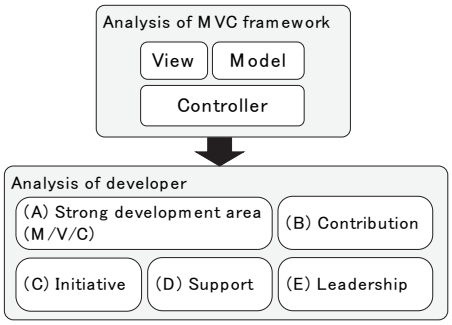

Figure 2. Items that can be analyzed using the proposed method

TABLE I. EXAMPLE OF ANALYSIS RESULTS FOR A SOFTWARE DEVELOPMENT PROJECT USING LARAVEL

```
----------controller------
all files: 57
Black Lotus: 12 ==> 21.05%
wu: 24 ==> 42.11%
leeke: 14 ==> 24.56%
fyy5: 1 ==> 1.75%
linmingkun: 6 ==> 10.52%

----------view----------
all files: 14
leeke: 5 ==> 35.71%
wu: 2 ==> 14.29%
Black Lotus: 7 ==> 50.0%

----------model----------
all files: 41
Black Lotus: 11 ==> 26.83%
wu: 15 ==> 36.59%
linmingkun: 9 ==> 21.95%
leeke: 5 ==> 12.20%
ttmk008: 1 ==> 2.44%
```

Table 1 shows an example set of analysis results for a software development project using Laravel, which is an MVC framework for PHP development. In this example, we can see that five developers are writing source codes in the controller layer, and the developer with the user name "wu" is the strongest contributor in this layer. Similarly, four and six developers are engaged in developing software components in the view and model layers, respectively; "Black Lotus" and "leeke" are the most active developers in the view layer, and "wu" and "Black Lotus" are the most active in the model layer.

From the above, we can assume that this project has three leaders: "wu", "Black Lotus", and "leeke". Each leader is contributing primarily to his/her strongest development areas (M/V/C) but sometimes provides support in other development areas. In addition, it is deemed that the developer "linmingkun" is a moderate contributor in the controller and model layers and that "fyy5" and "ttmk008" are supporting members of this project. Furthermore, we can see that the number of developers who are active in the view layer is apparently very few, and the burden for "Black Lotus" is very high in this layer. Therefore, for example, we can suggest that this project should seek additional developers in the view layer and allow "Black Lotus" to take over the second most active role in developing for the controller layer.

III. PROPOSED METHOD

A. Overview

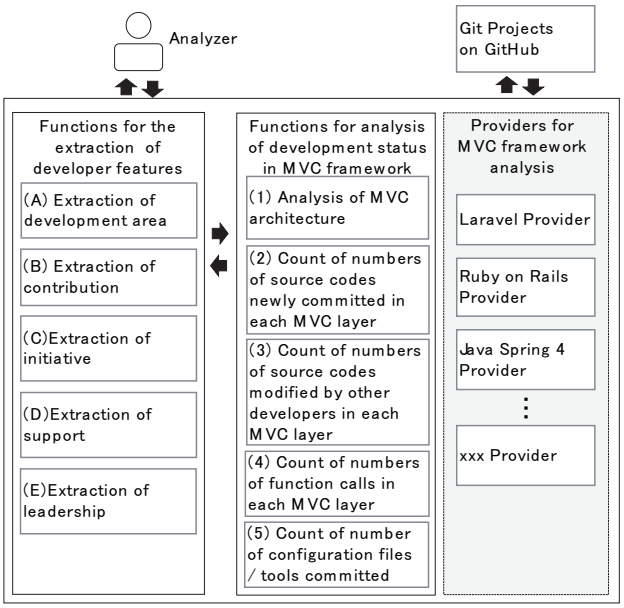

Figure 3. Overview of the proposed method

Figure 3 shows the architecture of our proposed method. The proposed method mainly consists of (1) functions for the extraction of software developer features, (2) functions for the analysis of the development status in the MVC framework, and (3) providers for MVC framework analysis. A provider for MVC framework analysis is a plugin module for analyzing the architecture of an individual MVC framework, such as Laravel, Ruby on Rails, or Java Spring 4. In addition, to analyze the development status, our method first analyzes the MVC architecture of the target project and then calculates the numbers of newly committed source

codes in each MVC layer, the numbers of source codes modified by other developers in each MVC layer, the numbers of function calls in each MVC layer, and the number of configuration files and tools committed for the construction of the project's development environment. Based on these data, our method extracts developer characteristics with regard to (A) development areas, (B) contribution, (C) initiative, (D) support, and (E) leadership.

B. Extraction of Developer Features

In this section, we describe our method of extracting developer characteristics with regard to (A) development areas, (B) contribution, (C) initiative, (D) support, and (E) leadership based on an MVC framework analysis.

[(A) Extraction of development areas]

Our method extracts the development areas in which each developer participates by counting the numbers of source code files that are created or modified at each development layer (controller, view, and model) in the MVC framework. For example, it is deemed that a developer who often creates source codes in the controller layer is assuming the role of a front-end engineer (Figure 4). When a developer has a creation and modification rate a for source code files in a given development layer that is greater than a threshold α%, our method assumes that this developer plays a major role in that development layer.

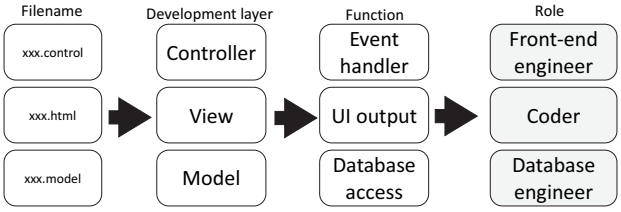

Figure 4. Extraction flow for development areas

[(B) Extraction of contribution]

As shown in Figure 5, our method extracts a developer's contribution level based on that developer's rate b of creating new source code files and modifying other developers' source code files (Table II). When a developer codes important functions such as library functions, the rate b is multiplied by a corresponding weight value w.

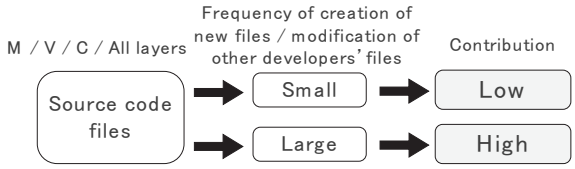

Figure 5. Extraction flow for contribution

TABLE II. CONTRIBUTION LEVEL

	Contribution level
$0 < b \leq \beta_1$	1
$\beta_1 < b \leq \beta_2$	2
$\beta_2 < b \leq 100$	3

[(C) Extraction of initiative]

For other developers on the project to be able to efficiently progress in their collaborative work, it is important for one or more developers to take the initiative in creating the configuration files, libraries, and management tools that are needed for the construction of the system environment (Table III). Therefore, our method assigns 'initiative' to a developer who often commits such programs and management tools (Figure 6).

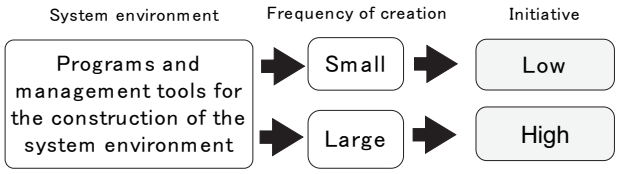

Figure 6. Extraction flow for initiative

TABLE III. EXAMPLES OF SOURCE PROGRAMS FOR THE CONSTRUCTION OF THE SYSTEM ENVIRONMENT

Name of management tool	Filename	Description
bower	bower.json	Management of javascript/css libraries on GitHub
npm	package.json	Management of nodejs libraries
grunt	gruntfile	Automatic processing for Node.js
gulp	gulpfile.js	Automatic processing for Node.js
composer	composer.json	Management of PHP libraries
pip	requirements.txt	Management of Python libraries
maven	pom.xml	Automatic construction for Java projects
gradle	build.gradle	Automatic construction for Java projects
gem	gemfile	Management of Ruby packages
godep	godeps.json	Management of libraries on which the Go language depends
cordova	config.xml	Development of smartphone applications using HTML5

[(D) Extraction of support]

Support means that, to some degree, a developer performs supportive activities for other developers on the project. Our method extracts 'support' for a developer when that developer's modification rate c of other developers'

source code files in a development layer (controller, view, or model) is greater than a threshold χ% (Figure 7).

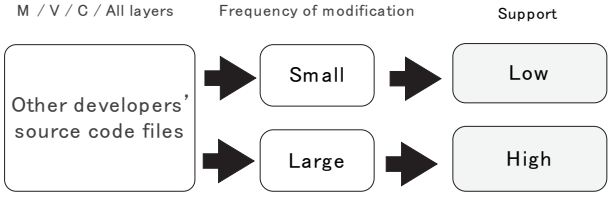

Figure 7. Extraction flow for support

[(E) Extraction of leadership]

Leadership is extracted based on a combination of a developer's characteristics in terms of (A) development areas, (B) contribution, (C) initiative, and (D) support, as described above. Figure 8 shows an example of leadership extraction based on initiative and support, when it is deemed that both of these characteristics are important factors for leadership on the considered project.

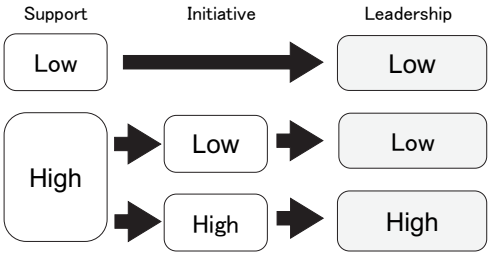

Figure 8. Example of an extraction flow for leadership

IV. EXPERIMENTS

To experimentally confirm the feasibility of our proposed method, we analyzed real Git projects on GitHub.

A. Experimental Environment

We analyzed four software development projects, P_1, P_2, P_3, and P_4, on GitHub, as described in Table IV. The MVC framework for each project was either Laravel for PHP or Ruby on Rails. The URL of project P_1 is not disclosed because this project is the author's own.

B. Experimental Method

We extracted developer characteristics with regard to (A) development areas, (B) contribution, (C) initiative, (D) support, and (E) leadership in projects P_1, P_2, P_3, and P_4 as described in Section III.

For (A), we set the threshold α% to 25% for projects P_1, P_2, and P_3 and to 1% for project P_4, because the number of developers on P_4 is very large (greater than 700).

For (B), we defined the contribution levels as follows: level 3 corresponds to a rate b of greater than 50%, level 2 corresponds to a b value of between 25% and 50%, and level 1 corresponds to a b value of less than 25%. Here, b is the rate of creation of new source code files and modification of other developers' source code files. For (C), we assigned 'initiative' to a developer when he/she committed programs or management tools one or more times. In addition, we set the support threshold c to 25% for projects P_1, P_2, and P_3 and to 5% for project P_4. Furthermore, we assigned 'leadership' to a developer when that developer exhibited the 'support' and 'initiative' characteristics.

TABLE IV. TARGET PROJECTS FOR ANALYSIS

ID	Project name (Framework name)	URL	# of analyzed developers / # of developers
P_1	Management system for chain convenience stores (Laravel)	Non-disclosed	6 / 6
P_2	Fullycms (Laravel)	https://github.com/sseffa/fullycms	2 / 2
P_3	Caravel-Blog (Laravel)	https://github.com/FbF/Laravel-Blog	3 / 6
P_4	Gitlabhq (Ruby on Rails)	https://github.com/gitlabhq/gitlabhq	16 / 753

C. Result

Figures 9 to 12 show the analysis results obtained by counting the numbers of source code files in each development layer (controller, view, and model) that were created or modified by each developer for projects P_1, P_2, P_3, and P_4. In addition, Tables V to VIII show the results of extracting developer characteristics based on these analysis results for projects P_1, P_2, P_3, and P_4. Based on these results, we can confirm that our method allows us to extract the characteristics of software developers in terms of "strong development areas (M/V/C)", "contribution", "initiative", "support", and "leadership" based on an MVC framework analysis of a Git project.

For example, in Figure 9, the rate of creation and modification of source code files by developer P1-6 is greater than 50%, and his contribution is very high. As a result, a contribution level of 3 is assigned to P1-6, as shown in Table V. In addition, P1-6 exhibits the 'support' characteristic and the 'initiative' characteristic because he has committed programs and management tools one or more times, as shown in Table IX; therefore, the 'leadership' characteristic is also assigned to P1-6. Meanwhile, developer P1-4 exhibits both the 'contribution' and 'support' characteristics; however, he has not committed files for the construction of the system environment. Therefore, the 'leadership' characteristic is not assigned to P1-4. Furthermore, because no development areas can be extracted for developer P1-2 from among the MVC layers and his contribution level is low, it is deemed that P1-2 is a new entrant to the project, who joined after the project had already progressed to some degree.

Because project P_1 in particular is the author's own project, we were able to confirm that the feature extraction results as shown in Table V are consistent with the actual characteristics of each developer on project P_1.

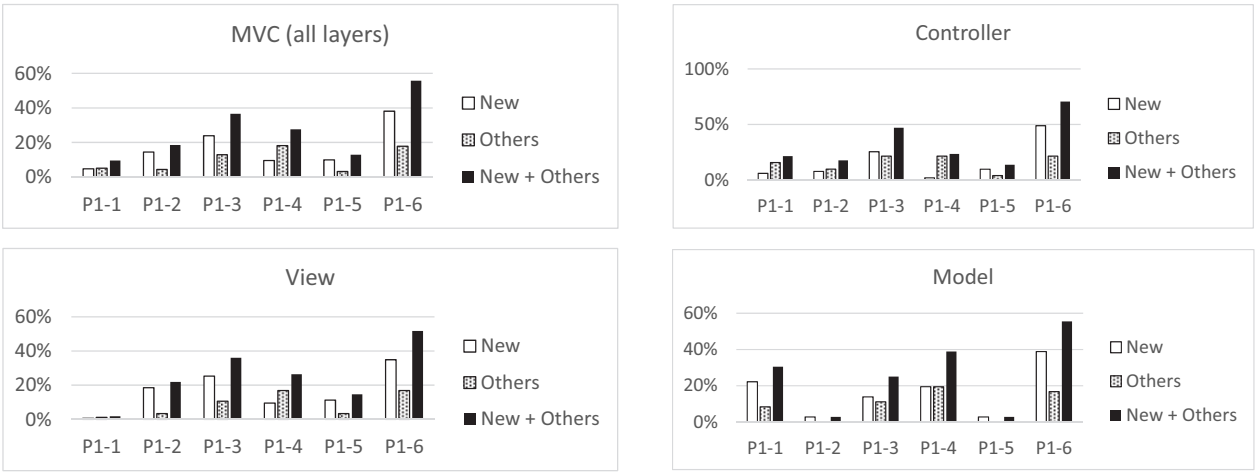

Figure 9. Analysis results for project P_1

Figure 10. Analysis results for project P_2

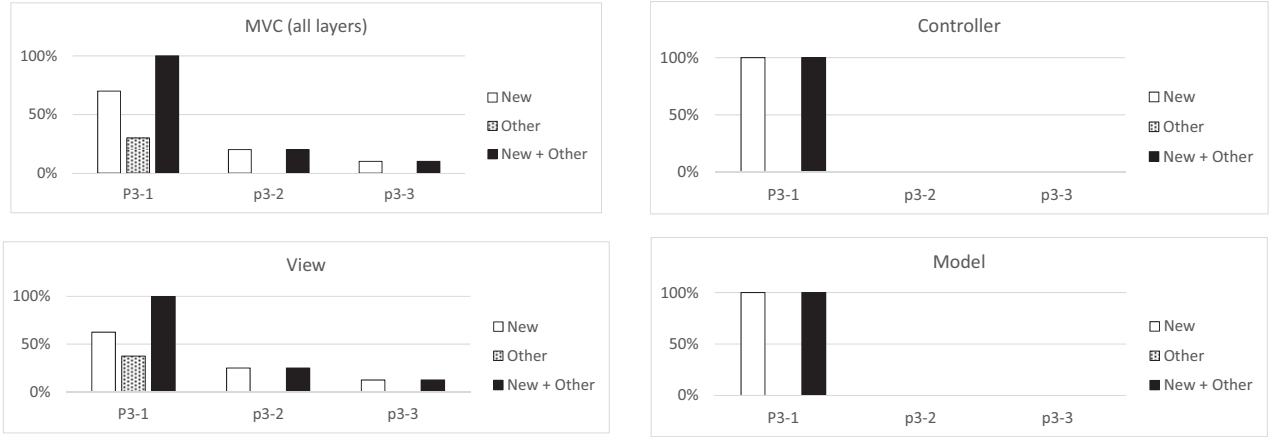

Figure 11. Analysis results for project P_3

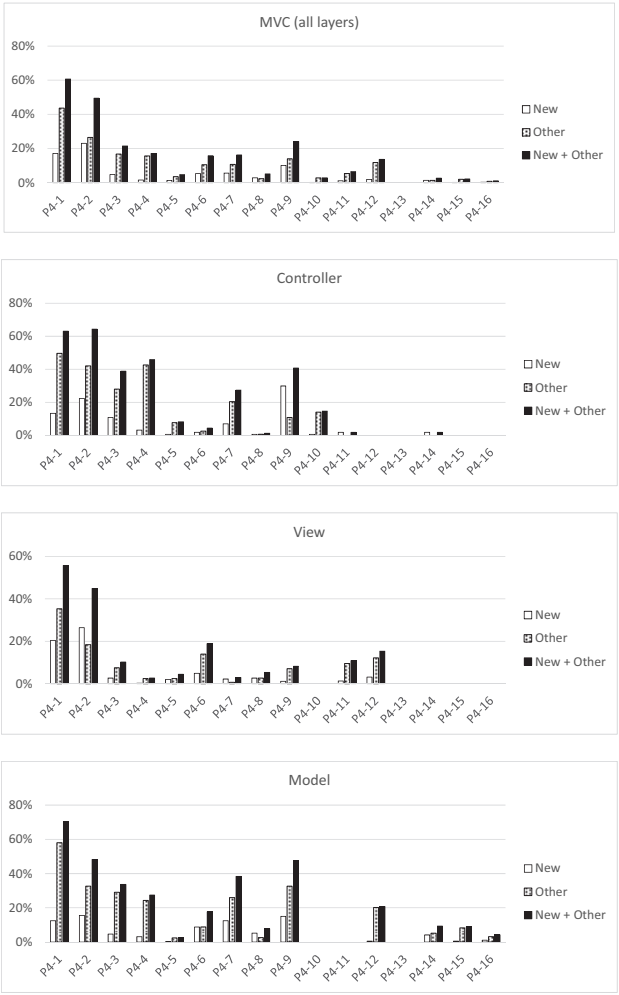

Figure 12. Analysis results for project P_4

TABLE V. FEATURE EXTRACTION RESULTS FOR DEVELOPERS ON P_1

Developer ID	(A) Develop-ment areas	(B) Contrib-ution	(D) Support	(E) Leader-ship
P1-1	CM	1	false	false
P1-2	-	1	false	false
P1-3	CVM	3	true	true
P1-4	CVM	3	true	false
P1-5	CV	2	false	false
P1-6	CVM	3	true	true

TABLE VI. FEATURE EXTRACTION RESULTS FOR DEVELOPERS ON P_2

Developer ID	(A) Develop-ment areas	(B) Contrib-ution	(D) Support	(E) Leader-ship
P2-1	CVM	3	true	true
P2-2	-	1	false	false

TABLE VII. FEATURE EXTRACTION RESULTS FOR DEVELOPERS ON P_3

Developer ID	(A) Develop-ment areas	(B) Contrib-ution	(D) Support	(E) Leader-ship
P3-1	VM	3	true	true
P3-2	V	2	false	false
P3-3	V	1	false	false

TABLE VIII. FEATURE EXTRACTION RESULTS FOR DEVELOPERS ON P_4

Developer ID	(A) Develop-ment areas	(B) Contrib-ution	(D) Support	(E) Leader-ship
P4-1	CVM	3	true	true
P4-2	CVM	3	true	true
P4-3	CVM	3	true	true
P4-4	CVM	3	true	true
P4-5	CV	3	false	false
P4-6	M	1	false	false
P4-7	CVM	3	true	true
P4-8	CVM	3	false	false
P4-9	CVM	3	false	false
P4-10	CV	3	true	true
P4-11	M	2	false	false
P4-12	VM	3	true	true
P4-13	V	3	true	true
P4-14	M	2	false	false
P4-15	M	2	false	false
P4-16	M	1	false	false

TABLE IX. COMMIT HISTORY OF SOURCE PROGRAMS FOR THE CONSTRUCTION OF THE SYSTEM ENVIRONMENT FOR P_1.

Management tool	Filename	Developer ID
bower (js/css manager tools)	bower.json	P1-6
composer (php)	composer.json	P1-6
gulp (nodejs)	gulpfile.js	P1-6
composer (php)	modules/Agent/composer.json	P1-1
composer (php)	modules/Area/composer.json	P1-1
composer (php)	modules/Common/composer.json	P1-3
composer (php)	modules/Master/composer.json	P1-3
composer (php)	modules/Shop/composer.json	P1-1
composer (php)	modules/Trade/composer.json	P1-6
composer (php)	modules/User/composer.json	P1-3
npm (nodejs package manager tools)	package.json	P1-6
bower (js/css manager tools)	public/master-static/bower.json	P1-3

V. CONCLUSION

In this paper, we presented a method of feature extraction for analyzing the characteristics of software developers based on a history of collaborative development using a distributed version control system. By means of experiments using real Git projects on GitHub, we confirmed that our method allows us to extract the characteristics of software developers with regard to "strong development areas (M/V/C)", "contribution", "initiative", "support", and "leadership" based on an MVC framework analysis of a Git project.

In our future work, we will design a more appropriate model of software developer behaviors that is suitable for actual collaborative development by evaluating developers who have participated in various Git projects. In addition, we will evaluate the effectiveness of our method by conducting several experiments with larger datasets concerning developers working on Git projects.

References

[1] GitHub API v3, https://developer.github.com/v3/, (visited on Jan. 13, 2016).

[2] Exploring Expressions of Emotions in GitHub Commit Messages, http://geeksta.net/geeklog/exploring-expressions-emotions-github-commit-messages/, (visited on Jan. 13, 2016).

[3] Ruby on Rails, http://rubyonrails.org/, (visited on Jan. 13, 2016).

[4] Spring, http://spring.io/, (visited on Jan. 13, 2016).

[5] Laravel, http://laravel.com/, (visited on Jan. 13, 2016).

[6] Saya Onoue, Hideaki Hata, and Ken-ichi Matsumoto, "A Study of the Characteristics of Developers Activities in Github," Proceedings of International Workshop on Empirical Software Engineering in Practice (IWESEP 2013), pp.7-12, 2013.

[7] Saya Onoue, Hideaki Hata, and Ken-ichi Matsumoto, "Developer Classification Based on Developers' Activities in GitHub [in Japanese]", Information Processing Society of Japan (IPSJ) Journal, Vol.56 (2), pp.715-719, 2015.

[8] Zhou, M. and Mockus, A. "Does the initial environment impact the future of developers?", Proceedings of the 33rd International Conference on Software Engineering, pp.271-280, 2011.

[9] Salleh, N., Mendes, E., Grundy, J., and Burch G.S.J., "An empirical study of the effects of personality in pair programming using the five-factor model", Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM '09), pp.214-225.

[10] Novielli, N., Calefato, F., and Lanubile, F., "The challenges of sentiment detection in the social programmer ecosystem", Proceedings of the 7th International Workshop on Social Software Engineering, pp.33-40, 2015.

[11] Guzman, E., Azócar, D., and Li, Y., "Sentiment analysis of commit comments in GitHub: an empirical study", Proceedings of the 11th Working Conference on Mining Software Repositories, pp.352-355, 2014.

[12] Stack overflow, http://stackoverflow.com/, (visited on Jan. 13, 2016).