University of Warsaw
Faculty of Economic Sciences

Working Papers
No. 15/2016(206)

DOROTA CELIŃSKA

WHO IS FORKED ON GITHUB? COLLABORATION AMONG OPEN SOURCE DEVELOPERS

Warsaw 2016

University of Warsaw
Faculty of Economic Sciences

Who is forked on GitHub? Collaboration among Open Source developers

Dorota Celińska
Faculty of Economic Sciences
University of Warsaw
e-mail: dcelinska@wne.uw.edu.pl

Abstract

In this article we investigate which characteristics of the developers involved in the creation of Open Source software favor innovation in the Open Source community. We utilize a unique database, obtained by web-scrapping GitHub from January to March, 2016. The results of the analysis show that higher reputation in the community improves up to a certain degree the probability of gaining collaborators, but developers are also driven by reciprocity, which is consistent with the concept of gift economy. There exists also a statistically significant network effect emerging from the standarization – developers using the most popular programming languages in the service are likely to have more collaborators. Providing additional contact information improves the chance of having coworkers. The obtained results can be generalized for the population of mature users of GitHub.

Keywords:

Open Source, GitHub, fork, collaboration, innovations, reputation, gift economy, network externality, standarization, reciprocity

JEL:

L15, L86, L17, L14, D85

1 Introduction

The Open Source software license allows the end users to study, modify, and distribute the publicly accessible source code to anyone and for any purpose. The creation of this kind of software usually relies on volunteer contributions and is associated with occurring *communities*, i. e. groups of people interested in or working on specific projects [32]. Communities form social networks, where innovations are created and distributed. By innovation, we mean "the process of commercialization of a newly developed or adopted product or practice" [12]. In particular, collaboration among developers which leads to the creation of a new product (new software, or improvements to previously existing one) can be viewed as a process which generates innovations.

The existing empirical research regarding Open Source software usually focuses on extracting the determinants of the Open Source license choice made on an enterprise or a project level [23, 3, 18, 9]. Additionally, a substantial literature revolves around explaining the incentives that motivate the individual developers to contribute to the Open Source projects [15, 36, 3, 20]. Economists focus on public good nature of this kind of software. Although Open Source products are also subjects to network externalities [3, 29, 7], it is a surprisingly rarely mentioned issue in the relevant literature.

Socially connected computing is rooted in the study of corporate portals and groupware [13]. A great amount of research has been conducted on examining SourceForge, an Open Source repository that started in 1999 [21, 11, 28]. SourceForge, despite its popularity, lacks features to make social ties and keeping up with other developer's updates [22, 6]. More recently implemented, GitHub is currently one of the largest repository hosting services related to the development of Open Source software, featuring elements of a social network service. As of Dec 31, 2015, GitHub repository hosting service has more than 16 million of registered users, and over 25 million of repositories. Although the registrations of the early users were in 2007, the service officially launched in April 2008.

The aim of this article is to investigatate which characteristics of developers involved in the creation of Open Source software favor innovation in the Open Source community. We analyze the volunteer developers. We will accomplish our objective by estimating the values of the parameters of logit model. The dependent variable will be the binary variable, indicating whether or not the collaboration with the particular developer happened (and the innovation was generated). We will utilize a unique database, obtained by web-scrapping GitHub.

This article is organized as follows. In Section 2 we give a short review of the existing empirical background. The GitHub's *fork&pull* model of collaboration is presented in Section 3. A brief characteristics of the social networks described in this study is provided in Section 4. The data set is presented in Section 5. In Section 6 we formulate the research hypotheses and introduce the empirical model, and the results of the conducted analysis are given in Section 7. The Section 8 provides discussion of the results and Section 9 concludes the research.

2 Factors promoting collaboration between Open Source developers

Innovations in the Open Source community are created through collaboration among developers working in a distributed environment [17]. Participation of developers in creating the publicly accessible source code is one of the popular topics in economic analysis of Open Source software [9, 33, 5]. Open Source software constitutes a kind of contradiction in terms of classical economic analysis: a special definition of ownership expressed in the license agreements and the common lack of direct remuneration for developers should discourage them from making effort, thus the source code should not be created. In fact, the behavior of developers is quite the opposite. The current empirical research has identified 3 types of factors encouraging developers to write the publicly accessible source code [9, 34]:

- *extrinsic motivation* – factors originating from the external sources: the environment in which the developers work;
- *intrinsic motivation* – psychological factors driving developers;
- *internalized extrinsic motivation* – factors that can be both external and psychological.

The results of empirical analyses pointed out that determinants of decision whether or not to take part in creating Open Source code include the increase in subjective value of created software and satisfying one's needs by adding a missing functionality [15, 23, 16, 35]. Having fun while coding and a potential boost in the professional career are also significant factors for developers [15, 23, 14, 21, 19]. According to Lerner and Tirole [23] developers decide to write publicly accessible source code, because they expect this to enhance their employment opportunities. Taking part in the Open Source

community and gaining visible popularity is treated as a kind of signal for the potential employers. The employers are aware of the existence of high quality Open Source projects that require excellent skills in programming, so if the given developer is mentioned in the log files of such projects, it can be treated as a proof of their skills and knowledge. The interest in developing the project arises also from the user's needs – lack of a necessary functionality encourages to provide it. Apart from economic factors, the developers of Open Source software are also driven by altruism [14, 35].

Having abundance of "survival necessities", like computing power and disk space in the Open Source community reveals this phenomenon's *gift economy* nature. Gift economy creates social structures and shapes the behavior of agents in case of excess instead of scarcity. In Open Source software case social status is related to "what you give away" and not to "what is under your control" [30, 26]. This leads to the situation in which reputation among developers' peers is the measure of competitive success. Several other studies also have shown that developers' anticipated increase in reputation (popularity in the community) resulting from the number of fixed bugs encourages them to take part in writing the Open Source code [15, 4, 31, 23]. Beside the reputation, due to the gift economy the expectations of the future gifts in return also shape the behavior of the developers. The reciprocity as a motivational factor for developers was reported in various studies [10, 1, 21].

The diverse motivational components are not necessarily mutually exclusive and may co-exist within a developer. The main focus of this stream of literature is to understand how developers' motivations drive their participation/effort on Open Source project which in turn affects the overall performance and effectiveness of developers and projects.

3 Github's *fork&pull* model of contributing to the source code

Every registered user of GitHub has their own site that integrates social media functionality directly with code management tools [25]. Each individual user's profile contains public information about their biographical characteristics (the date of joining the service and the optional description of location, employer, personal site and e-mail address), the list of public project repositories, and the number of people following the user as well as the number of people that the user follows. Everyone, even unregistered users of the service may browse the profiles and download public project

repositories.

The contribution to the project may occur in at least two ways [6]:

- by sending via e-mail the patches to the author of the original project;
- by submitting the *pull requests* – the proposed changes to the projects that may be accepted or declined by the maintainers of the project.

The difference between those two options is that the submitting of pull request requires the prior registration in the repository hosting service. The developer that is willing to commit to the original repository (and thus make it an upstream repository commit) has to *fork* the existing project repository. Forking means creating developer's own copy of the source code. As a result the user (according to the GitHub's jargon now known as a *member*) is granted the access to the complete source code. The access also includes all branches that were created before the project had been forked. The members may introduce their commits (proposed bug fixes and changes to the source code) locally, but are also allowed to synchronize the state of their repositories with the state of the original project. If the maintainer of the original project repository accepts the changes introduced in the pull request, both repositories are merged and the history of submitted commits by the individual users is preserved.

4 Collaboration and reputation emerging from GitHub's social networks

GitHub is a collaborative repository hosting service that includes social features bringing a new transparency to the development project [27]. The activity of its registered users forms several kinds of social networks. The most intuitive one is the network of collaboration between developers within the project repositories. The existence of such a network stems directly from the Open Source licenses' statements, that enable modifications of the publicly accessible source code. Due to the special model of collaboration in the GitHub repository hosting service, this network is embedded in the observable network of users that are granted permission to contribute to the projects – members. Collaboration among GitHub users can be seen in at least two ways. Firstly, one can analyze bipartite graphs of developers and their particular repositories they contribute to. Secondly, as suggested by Lima et al. [24], one can project the mentioned graph onto the set of users – this way, developers who contribute to at least one common repository are

connected to each other. Moreover, we utilize the third way of mapping the relationship of collaboration: the connection between the initial owner of the project repository and the members of the repository (however, the members themselves are not connected). This way we preserve the hierarchical nature of the most popular model of collaboration in GitHub.

The second type of social network that can be observed in the GitHub repository hosting service is the network of *followers* – users who agreed to be sent notifications about one's activity on GitHub. The network of followers may be seen as a proxy of influence and reputation process in the service. As Goggins and Petakovic [13] state: following a developer gives them a degree of influence. The more followers a developer has, the larger the group potentially interested in their work, creating a potential for even greater influence. Even if not capable of contributing to the projects of "globally" popular developers, users of GitHub are interested in their work. This explains enormous numbers of followers of celebrities, like GitHub's staff. On the other hand, following less popular developers improves the efficiency of matching possible coworkers. This is closely related to the *spillover effect*. Knowledge spillovers are reported by Fershtman and Gandall [11] to be correlated with the structure of the social networks, since contributors who work on several projects are likely to exchange information and knowledge. While following a person, one is alerted about their activity in the service: the creation of new repositories, submitted commits, issues and pull requests, along with starring (giving a distinction to) another developer's project repository. The process of following should considerably decrease the cost of acquiring information about the possible coworkers and the projects one would like to contribute to, since developers are quickly given the notification instead of browsing the service on their own. What is more, the analysis of 199 GitHub's most followed users and their followers by Blincoe et al. [2] showed that popular users influence their followers by guiding them to new projects.

Similar to the network of followers, yet of different kind of information provided, is the network of watchers. A watcher is a follower of the particular repository. While notifications stemming from following a developer supply quite a diverse range of information, the alerts for watchers are confined to the activity within one repository (mostly issues, bug fixes, and the collaborations within project). Intuitively, watchers have probably been already interested in the development of that particular project, but one may also see the process of gaining new watchers as another proxy for reputation of the developer.

5 Data set

The data was collected using the web-scrapping technique, i.e. automatic extraction of information from websites by web crawlers – Internet bots which systematically browse the World Wide Web. Scrapping is focused on the transformation of unstructured data on the web, typically in HTML format, into structured data that can be stored and analyzed, e.g. in a spreadsheet. The data about registered users and their repositories is publicly available but very distributed. We have collected the snapshots of the networks observed in GitHub repository hosting service from January to March, 2016.

The population of this research consists of users of GitHub repository hosting service registered in 2007-2011, which means 1 296 987 entities. We limited the span of registrations to make sure the developers had a chance to gain popularity, learned how to use the service and had enough time to start collaborating with others. Similarly to Lima et al. [24], our sample is smaller (553 370 observations): it consists only of registered users that are active (did not delete their profiles) and have at least one public repository. We disregard the "organization" accounts due to the non-random missing data patterns connected with those profiles.

6 Methodology

We consider a group of agents (developers) $\mathcal{V} = \{1, \ldots, N\}$, who are members of a social network represented by a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where the set of edges $\mathcal{E}$ represents connections between agents. Self-loops are not allowed in this graph. Each agent (a node in the graph) is described with a set of characteristics, such as number of repositories hosted on GitHub, number of stars (distinctions) given to their projects, number of followers, number of followed developers, number of watchers, etc.

In this study we consider three types of social networks. We represent the relation of forking one's project repository (and thus creating an innovation by collaboration with the owner of the repository) by means of directed graph $\mathcal{G}_m$, which we call the members graph. The edges of this graph show the hierarchical structure of forking: there is an edge in $\mathcal{G}_m$ between A and B if and only if A is a member of a project repository owned by B. The second directed graph is the followers graph $\mathcal{G}_f$ representing the following relations among developers. There is an edge in $\mathcal{G}_f$ between A and B if and only if A follows B. The third social network described in this study is the

bipartite graph of developers and the repositories they own: the repository graph $\mathcal{G}_r$.

6.1 Model, design and research hypotheses

We utilize a logit model. Logit models form the family where the dependent variable is categorical. The binary logit model is used to estimate the probability of a binary response based on one or more independent variables [8]. In our case, the dependent binary variable denotes whether a particular developer was forked (and thus the inovation by collaboration with them was generated) or not. This is equal to the dummy indicating in-degree of the node in $\mathcal{G}_m$ greater than 0. We divide our independent variables into a set of five types:

- **Reputation**: the number of developer's *followers* (in-degree of a node in $\mathcal{G}_f$), *stars obtained* by a developer and the number of developer' *watchers* (respectively, the sum of stars and the sum of watchers of developer's repositories in $\mathcal{G}_r$);
- **Attitude** toward others: the number of developers *followed* by a developer (out-degree of a node in $\mathcal{G}_f$), a dummy denoting *forking* or not other developers (1 being the nodes with out-degree in $\mathcal{G}_m$ greater than 0), and *stars given* to other developers' repositories;
- **Standarization**: a set of dummies indicating whether developer owns a project written in one of the 13 most popular *languages* in the service (constructed upon the characteristics of developer's repositories in $\mathcal{G}_r$);
- **Information**: a set of dummies indicating whether developer provides a valid *e-mail* address or a valid url to their personal *site*;
- **Control**: the number of *repositories* (the out-degree of the nodes in $\mathcal{G}_r$), the *year* of developer's registration.

We will verify following hypotheses:

Hypothesis 1 *The reputation proxies significantly and positively affect the probability that others would like to contribute with a particular developer. However, this impact is nonlinear.*

Hypothesis 2 *The attitude proxies significantly and positively affect the probability that one would have the coworkers (one of the motivational factors for developers is reciprocity).*

Hypothesis 3 *There exists a significant network effect emerging from standarization: the users of the most popular programming languages tend to be more likely to have collaborators.*

Hypothesis 4 *Developers who provide additional contact information publicly are more probable to be forked. The contact information means e-mail address or url of the personal site.*

7 Results

The results of the conducted analysis in the form of the values of coefficients of logit model are presented in Table 1.

To validate the functional form of the model, we performed the linktest. The result turned out to be statistically insignificant (p-values > 0.05), which means that the functional form of our model is correct.

The coefficients for the independent variables from the *reputation* set were always statistically significant. The significance of the squares of the variables supports Hypothesis 1 about the non-linear impact of reputation proxies on the probability of collaborating with a developer. Intuitively: developers with excellent programming skills are likely to be highly rewarded in the community, but their projects may be at the same time characterized by such high entrance costs and the degree of specialization that they discourage other developers from collaboration. Thus the reputation proxies evince diminishing returns to scale. The only exception is the case of *followers* related variables, which show increasing returns to scale. But this results is consistent with the result obtained by Celińska [6] and Lima et. al. [24] about the power-law scaling behavior of the $\mathcal{G}_f$. The results regarding reputation are similar to those described in the existing literature [15, 21, 23].

The obtained results for the *attitude* set of variables support Hypothesis 2. Developers are driven by reciprocity as reported in [10, 1, 21], but only up to a certain degree. Exaggeration while giving distinctions to others may lead to perception of a diminishing value of those distinctions. Such behavior in connection with lower reputation may also indicate the users with lower programming skills, which in turn may discourage from collaboration with them.

The coefficients of the programming language related variables were always statistically significant. Having a project repository written in one of the most popular languages in the service increases the probability of finding

new collaborators. This supports Hypothesis 3 that the Open Source software is characterized by a network effect derived from specialization. This kind of externality can be also seen as the *lock-in effect*.

The obtained results support Hypothesis 4. People providing valid e-mail adresses and urls to their personal sites reduce the cost of obtaining information about them. It also helps in forming one's impression in the service which corresponds with the suggestions of Marlow et. al. [25].

8 Discussion and further research

Due to the power-law scaling behavior characterizing networks emerging from GitHub as reported by Lima et. al. [24] and Celińska [6] the results of this study are representative for the population of active users of GitHub. However, the utilized data set contains mostly information about mature users of the service. This is one of the major limitations of the current study. We lack the comparison with the possible "newcomers", e.g. users who have recently registered.

We are fully aware that our data set suffers a time bias: we are not able to collect data about the whole repository hosting service in a preferably short time. The data utilized in this study is a snapshot of the service, the exact networks may differ. However we do not find this observation a threat to the obtained results: rare random missing observations should not impact the correlation structure, which was the topic of our interest.

This study focuses on determining the factors enhancing one's probability of gaining collaborators in GitHub. In particular, we investigate the characteristics of the graph nodes. A competitive approach basing on the characteristics of dyadic data sets would provide further insight into the determinants of collaboration among developers. This problems is equal to the problem of the occurance of edges in the graph. We suppose that the set of variables utilized in this study is not complete: further studies would include e.g. the gender aspects and the spatial analysis of the collaboration in the service.

9 Conclusion

The analysis of the relationships among the Open Source software developers is a rare subject of the relevant literature. In this article we have shown that higher reputation in the community improves up to a certain degree the probability of gaining collaborators, but developers are also driven by

reciprocity. This is consistent with the gift economy concept. There exists also a statistically significant network effect emerging from the standarization. Providing additional contact information also improves the chance of having collaborators.

This analysis differs from the existing research in various dimensions. Firstly, we propose a different specification of the collaboration network, preserving its underlying hierarchical structure. We also limit the population of the users, to assure that they had enough time to gain popularity and collaborate with other developers. Furthermore, we apply statistical tests and combine the social network analysis with econometric model to quantitatively describe the phenomenon. The obtained results can be generalized for the population of mature users of GitHub.

References

[1] Magnus Bergquist and Jan Ljungberg. The power of gifts: Organizing social relationships in open source communities. *Information Systems Journal*, 4(11):305–320, 2001.

[2] Kelly Blincoe, Jyoti Sheoran, Sean Goggins, Eva Petakovic, and Daniela Damian. Understanding the popular users: Following, affiliation influence and leadership on github. *Information and Software Technology*, 70:30–39, 2016.

[3] Andrea Bonaccorsi and Cristina Rossi. Why open source software can succeed. *Research Policy*, 32(7):1243–1258, 2003.

[4] Andrea Bonaccorsi and Cristina Rossi. Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. *Knowledge, Technology and Policy*, 18(4):40–64, 2006.

[5] Dorota Celińska. Użycie oprogramowania Open Source co poza gift economy? *Ekonomia journal*, 37, 2014.

[6] Dorota Celińska. Information and influence in social network of open source community. Unpublished manuscript, 2016.

[7] Dorota Celińska and Miroslawa Lasek. Why do users choose Open Source software? Analysis of the network effect. Working Papers 2015-05, Faculty of Economic Sciences, University of Warsaw, 2015.

[8] David R. Cox. The regression analysis of binary sequences (with discussion). *J Roy Stat Soc B*, 20:215–242, 1958.

[9] Kevin Crowston, Kangning Wei, James Howison, and Andrea Wiggins. Free/libre open-source software development: What we know and what we do not know. *ACM Comput. Surv.*, 44(2):7:1–7:35, March 2008.

[10] Paul David and Joseph Shapiro. Community-based production of open source software: What do we know about the developers who participate? Discussion Papers 08-003, Stanford Institute for Economic Policy Research, 2008.

[11] Chaim Fershtman and Neil Gandal. Microstructure of collaboration: The 'social network' of open source software. CEPR Discussion Papers 6789, C.E.P.R. Discussion Papers, 2008.

[12] Chris Freeman and Luc Soete. *The Economics of Industrial Innovation, 3rd Edition*, volume 1. The MIT Press, 3 edition, 1997.

[13] Sean Goggins and Eva Petakovic. Connecting theory to social technology platforms: A framework for measuring influence in context. *American Behavioral Scientist*, 58(10):1376–1392, 2014.

[14] Il-Horn Hann, Jeff Roberts, and Sandra Slaughter. Why developers participate in open source software projects: an empirical investigation. *International Conference on Information Systems 2004*, 2004.

[15] Alexander Hars and Shaosong Ou. Working for free? motivations for participating in open-source projects. *Int. J. Electron. Commerce*, 6(3):25–39, April 2002.

[16] Guido Hertel, Sven Niedner, and Stefanie Herrmann. Motivation of software developers in open source projects: an internet-based survey of contributors to the linux kernel. *Research Policy*, 32(7):1159–1177, 2003.

[17] Bruce Kogut and Anca Metiu. Open-source software development and distributed innovation. *Oxford Review of Economic Policy*, 17(2):248–264, 2001.

[18] Heli Koski. Private-collective software business models: Cordinatitons and commercialization via licensing. Discussion Papers 1091, The Research Institute of the Finnish Economy, 2007.

[19] Sandeep Krishnamurthy. On the intrinsic and extrinsic motivation of free/libre/open source developers. *Knowledge, Technology and Policy*, 18(4):17–39, 2006.

[20] Sandeep Krishnamurthy, Shaosong Ou, and Arvind K. Tripathi. Acceptance of monetary rewards in open source software development. *Research Policy*, 43(4):632–644, 2014.

[21] Karim Lakhani and Robert Wolf. *Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects*. MIT Press, Cambridge, 2005.

[22] Michael Lee, Bruce Ferweda, Junghong Choi, Jungpil Hahn, Jae Yun Moon, and Jinwoo Kim. Github developers use rockstars to overcome overflow of news. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, pages 133–138, 2013.

[23] Josh Lerner and Jean Tirole. Some simple economics of open source. *Journal of Industrial Economics*, 50:197–234, 2002.

[24] Antonio Lima, Luca Rossi, and Mirco Musolesi. Coding together at scale: Github as a collaborative social network. *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.

[25] Jennifer Marlow, Laura Dabbish, and Jim Herbsleb. Impression formation in online perrs production: Activity traces and personal profiles in github. In *Proceedings of 2013 Conference on Computer Supported Cooperative Work*, pages 117–128, 2013.

[26] Graziella Marzi. If not for money for what? Digging into the OS/FS contributors motivations. Working Papers 166, University of Milano-Bicocca, Department of Economics, July 2009.

[27] Nora McDonald, Kelly Blincoe, Eva Petakovic, and Sean Goggins. Modelling distributed collaboration on github. *Advances in Complex Systems*, 17(07n08):14500–14524, 2014.

[28] Regis Meissonier and Isabelle Bourdon. Toward an enacted approach to understanding oss developers motivations. *International Journal of Technology and Human Interactions*, 8(2):38–54, 2012.

[29] Ioana Popovici. The determinants of open source quality: An empirical investigation. Working Papers 704, Florida International University, 2007.

[30] Eric Raymond. Homesteading the noosphere. *First Monday*, 3(10), 1998.

[31] Eric S. Raymond. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2001.

[32] Eric S. Raymond. *The Art of UNIX Programming.* Pearson Education, 2003.

[33] Aaron Schiff. The economics of open source software: A survey of the early literature. *Review of Network Economics*, 1, 2002.

[34] Georg Von Krogh, Stefan Haefliger, Sebastian Spaeth, and Martin W. Wallin. Carrots and rainbows: Motivation and social practice in open source software development. *MIS Q.*, 36(2):649–676, June 2012.

[35] Chorng-Guang Wu, James H. Gerlach, and Clifford E. Young. An empirical analysis of open source software developers' motivations and continuance intentions. *Information & Management*, 44(3):253–262, 2007.

[36] Yunwen Ye and Kouichi Kishida. Toward an understanding of the motivation open source software developers. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 419–429, Washington, DC, USA, 2003. IEEE Computer Society.

Table 1: The results of the logit model

Set	Variable	Estimate	Std. error	z-value	p-value	
Reputation	followers	5.420e-03	3.799e-04	14.266	< 2e-16	***
	followers^2	4.667e-06	2.470e-07	18.895	< 2e-16	***
	watchers	4.060e-02	5.573e-04	72.850	< 2e-16	***
	watchers^2	-1.256e-05	1.363e-06	-9.217	< 2e-16	***
	stars_obtained	1.134e-02	2.019e-04	56.169	< 2e-16	***
	stars_obtained^2	-5.446e-06	8.341e-08	-65.290	< 2e-16	***
Attitude	following	4.073e-03	2.944e-04	13.836	< 2e-16	***
	following^2	-2.510e-08	1.824e-09	-13.761	< 2e-16	***
	stars_given	1.848e-05	3.680e-05	0.502	0.6156	
	stars_given^2	-3.721e-08	4.710e-09	-7.901	2.76e-15	***
	forking_others	1.667e-02	8.027e-03	2.076	0.037858	**
Standarization	langCSS	2.652e-01	9.766e-03	27.154	< 2e-16	***
	langHTML	3.006e-01	1.097e-02	27.415	< 2e-16	***
	langJava	4.442e-01	7.720e-03	57.543	< 2e-16	***
	langPHP	3.262e-01	8.673e-03	37.615	< 2e-16	***
	langPython	4.177e-01	8.105e-03	51.538	< 2e-16	***
	langRuby	2.062e-01	8.373e-03	24.627	< 2e-16	***
	langShell	4.576e-01	9.887e-03	46.285	< 2e-16	***
	langC	4.645e-01	9.602e-03	48.373	< 2e-16	***
	langC#	3.636e-01	1.197e-02	30.369	< 2e-16	***
	langCPP	4.510e-01	9.889e-03	45.609	< 2e-16	***
	langObjectiveC	8.909e-02	1.199e-02	7.432	1.07e-13	***
	langJavaScript	4.184e-01	7.728e-03	54.144	< 2e-16	***
	langGo	7.781e-02	1.609e-02	4.837	1.32e-06	***
Information	e-mail	1.785e-01	7.390e-03	24.158	< 2e-16	***
	site	1.715e-01	7.807e-03	21.965	< 2e-16	***
Control	year2009	-1.644e-01	1.748e-02	-9.407	< 2e-16	***
	year2010	-1.175e-01	1.639e-02	-7.172	7.37e-13	***
	year2011	-4.330e-01	1.592e-02	-27.193	< 2e-16	***
	repositories	-2.663e-03	6.985e-04	-3.813	0.000137	***
	repositories^2	-1.373e-04	2.994e-06	-45.854	< 2e-16	***
	(Intercept)	-2.077e+00	1.735e-02	-119.729	< 2e-16	***

Source: Own calculations made in R.

* denotes variable significant at significance level 0.1

** denotes variable significant at significance level 0.05

*** denotes variables significant at significance level 0.01

Faculty of Economic Sciences
University of Warsaw
44/50 Długa St.
00-241 Warsaw
www.wne.uw.edu.pl