Understanding "Watchers" on GitHub

Jyoti Sheoran, Kelly Blincoe, Eirini Kalliamvakou, Daniela Damian, Jordan Ell
Software Engineering Global Interaction Lab
University of Victoria
Victoria, BC, Canada
jsheoran@uvic.ca, kblincoe@acm.org, ikaliam@uvic.ca, danielad@cs.uvic.ca, jell@uvic.ca

ABSTRACT

Users on GitHub can watch repositories to receive notifications about project activity. This introduces a new type of passive project membership. In this paper, we investigate the behavior of watchers and their contribution to the projects they watch. We find that a subset of project watchers begin contributing to the project and those contributors account for a significant percentage of contributors on the project. As contributors, watchers are more confident and contribute over a longer period of time in a more varied way than other contributors. This is likely attributable to the knowledge gained through project notifications.

Categories and Subject Descriptors
D.2.9 [**Software Engineering**]: Management – *programming teams.*

General Terms
Management, Human Factors, Languages.

Keywords
GitHub, Repositories, Watchers, Software Teams.

1. INTRODUCTION

Open source software projects are initiated and maintained by a community of motivated developers that collectively create software. However, their contribution is often voluntary, and projects must sustain a pool of interested developers to receive contributions. Open source software development teams often exhibit a layered structure consisting of groups of developers varying in size and contribution focus. Core contributors are the smallest group and focus on building the codebase. Peripheral contributors are more active in bug reporting and submitting patches. There are also groups of active users that do not contribute code but provide feedback on new releases and undertake supporting activities [1]. Project success is related to the size of the developer community around it, which acts as a pool of potential contributors [2]. Without a renewable source of contributors, an open source software project can stagnate or fail.

GitHub is a popular code-hosting site that hosts a large number of open source software projects. It provides social features that allow for a community of contributors to be built around the codebase. Any user can "watch" a public repository to receive notifications about events in a repository such as new commits, pull requests, and issues. "Watching" a repository signals interest in the repository's activity and a potential interest in contributing [3]. It can be seen as a passive type of project membership.

In August 2012, GitHub changed how "watching" works [4]. "Starring" replaced the existing watching functionality, and any repositories a user had previously watched now appeared in their stars list. Starring allows users to mark repositories of interest to them, but they do not receive notifications about those repositories. At the same time, the new watching functionality was introduced which provided more detailed notifications on repositories. A user, therefore, who has chosen to watch a repository over simply starring the repository, has shown an interest in knowing the details of the activity involved in that project. Receiving such detailed notifications about a repository requires a time investment on the part of the watcher to read through these notifications and understand the project activity.

To explore whether watchers on open source software projects are potential contributors that will sustain the project's evolution, we investigate whether watchers eventually contribute to the projects they watch. We are guided by the following research questions:

RQ1: *Do watchers become contributors?* We look for evidence of project watchers taking a more active role on the project by contributing to the project. We examine the following contribution types: issue reporting, issue assignment, commenting on issues, creating pull requests, commenting on pull requests, committing code, and commenting on commits.

RQ2: *Do watchers behave differently than other contributors?* We investigate the first type of contribution, length of contribution and variety of contribution for both watchers and other contributors to look for different patterns of behavior between the two groups.

RQ3: *Does a project's programming language impact how its watchers behave?* We investigate whether watchers are more likely to contribute to projects based on the project's programming language. We also examine differences in the type of the first contribution and the length of time it takes watchers to begin contributing by programming language.

2. RELATED WORK

Watchers on GitHub introduce a new type of passive project membership. Very little research has investigated watchers on GitHub. Through developer interviews, Dabbish et al. found that GitHub users consider the number of watchers on a project as a signal of community importance and project quality [3]. Users also learn of interesting projects by considering the watching activity of other, influential users [3][5]. Marlow et al. found that projects with a greater number of users authorized to make changes directly to the main branch without approval also have a larger number of watchers [6]. However, the limited research on watchers has not considered the influence of watchers to the project through contribution activity. Our work addresses this gap.

3. RESEARCH METHODOLOGY

We used the MSR 2014 Mining Challenge MySQL Dataset [7], which contains data on 91 GitHub repositories and their forks. Of these, we analyzed the 72 projects that include information on watchers. For each project, we included data from the main branch and its forks [8]. The dataset contains user information from 19-Oct-2007 to 10-Oct-2013. During this time, 55,265 users contributed and 141,300 users began watching a project.

We collected contribution related information for each contributor on each project. We identified seven different contribution types: writing code, creating pull requests, reporting an issue, assigning an issue, commenting on commits, commenting on issues and commenting on pull requests. For each contribution type, we obtained the timestamp when the contribution was made.

GitHub commits have two associated users - author and committer. The author is the user who writes the code, while the committer commits the code. In 90% of commits, the author and the committer are the same user. We consider the author as the contributor since they are the one who completed the work.

GitHub introduced a Fork & Pull model for code contribution. Anyone can create a fork of a repository. Forking is similar to creating a local copy of the repository. New contributors who do not have access to the main repository can fork the repository and make code changes to the local (forked) project. A pull request is then created to alert the maintainers of the main repository that the fork owner has changes that he would like to be pulled into the main repository. Typically, a code review is performed prior to the change being merged into the main repository. Prior to GitHub, forks had a negative connotation among developers, but they are now a measure of a project's popularity.

4. RESULTS

We ran Pearson correlations between the number of watchers and the number of issues, forks, and commits for each project (results in Table 1). Pearson correlations were used because these measures are normally distributed. Projects that are popular, measured by fork count, attract more watchers. This is not surprising since forking a project shows an interest in that project, and 46% of users who fork a project are also watching the project. More active projects, measured by the number of issues, also have more watchers. However, the number of commits, another measure of project activity, does not correlate with the number of watchers. This suggests that watchers may be more interested in project issues than of the details of code changes.

Table 1. Correlations between Watchers and Project Activity.

Project Statistic	**Pearson Correlation to Number of Watchers**
Number of Forks	0.81***
Number of Issues	0.47***
Number of Commits	0.07***

(* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$)

RQ1: Do watchers become contributors?

Many projects have a large pool of watchers. A small percentage of these watchers ultimately contribute to the project in some way. Across all projects, 4.7% of watchers later became contributors. This ratio is fairly consistent across the individual projects. The mean percentage of watchers who become contributors on an individual project is 5.4%, with a median of 5.0%. However, while the ratio of watchers who become contributors (4.7%) is not large, those watchers account for a large portion of the total population of contributors (20.7%). This is consistent across individual projects (mean = 22.7% and median = 22.4%).

On average, watchers start contributing after 55 weeks of watching, (median = 48). However, this varies greatly depending on the type of their first contribution. Table 2 shows the average duration between becoming a watcher and the watcher's first contribution for each contribution type. Surprisingly, commenting on a commit has the quickest turnaround, which can be viewed as a type of code review. While commenting on pull requests, another type of code review, takes nearly twice as long.

Table 2. Average Duration between Becoming Watcher and First Contribution by First Contribution Type.

First Contribution Type	Mean Time Before First Contribution (weeks)
Comment on Commit	36.7
Assigned Issue	37.3
Submit Code	51.8
Report Issue	56.1
Create Pull Request	61.4
Comment on Pull Request	63.1
Comment on Issue	91.1

Answer to RQ1: While the percentage of watchers who became contributors is not large, their number in the total contributor population is significant. This indicates that having a large pool of watchers is important to recruiting the project's future contributors and to the health of the project.

RQ2: Do watchers behave differently than other contributors?

Watchers who became contributors are more confident than other contributors, based on their first contribution type. 87.6% of watchers who contribute start contributing by reporting issues, submitting code and creating pull requests. These types of contributions are likely the most valuable to open source projects when compared to comments on issues and comments on pull requests. Only 56.2% of other contributors start with these types of contributions. Contributors who were not first watchers are most likely to start contributing by simply commenting on an issue. This type of action is likely to require the least amount of confidence since many comments on issues just note agreement with the described issue. On the other hand, comments on commits are likely to describe a problem with the code [3], and a contributor must feel very confident in their opinion to critique the work of others so publicly. Contributors who were watchers first are much more likely to make a commit comment as their first contribution. Table 3 shows the first contribution type for watchers and other contributors. Chi-squared tests show that for all contribution types except code submission, the difference in proportion between the watchers and other contributors is significant. The table excludes issue assignment because it accounts for less than 0.1% of first contributions.

Besides being more confident in their first contribution, watchers are also more likely to have sustained contribution than other contributors. Watchers contribute for 16.4 weeks on average, whereas other contributors contribute for only 7.5 weeks on average, where contribution duration is computed as first contribution to last contribution. However, the contribution duration is not normally distributed; the median for both groups is 0 weeks. The median number of contributions from both groups of contributors is one, showing that "drive-by" [9] contributions are common. However, a Mann-Whitney test shows that the distribution of contribution durations is significantly different between watchers and other contributors (W= 279824330, $p<0.001$), but there is not a significant difference in the number of contributions between the two groups. This shows that while watchers may not contribute more than other contributors, their vested interest in the project through receiving notifications causes them to contribute over a longer time period.

Watchers are more likely to provide more varied contributions than other contributors. A Mann-Whitney tests shows that the number of contribution types per contributor is significantly greater for contributors who were watchers first compared to other contributors (W = 299586686, $p < 0.001$).

Table 3. First Contribution.

Contribution	Watchers	Others	Chi-Squared Test
Report an issue	43.9%	24.5%	2042.5***
Submit code (commit)	26.3%	27.5%	7.6***
Pull requests	17.4%	4.2%	3028.4***
Comment on Issue	2.1%	39.4%	7117.7***
Comment on Commit	8.7%	4.1%	485.6***
Comment on pull request	1.2%	0.2%	267.8***

(* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$)

Table 5 shows the contributors second type of contribution based on the type of their first contribution. 65.45% of watchers whose first contribution type is code submission (either on the main branch or on their own fork) go on to submit a pull request. However, only 12.01% of other contributors follow up with a pull request after submitting code. Other contributors often comment on an issue after submitting code, indicating that they perhaps test a code solution on their own fork prior to making a comment.

Contributors who commented on a commit or a pull request as their first contribution are more likely to submit code in the future. Perhaps they found the code they commented on problematic enough that they felt compelled to submit their own solution.

40.1% of watchers contribute in another type of activity (contribution type), whereas only 16% of other contributors later perform a different type of contribution. Table 4 shows the percent of contributors who perform a different type of contribution from their first contribution type for watchers and other contributors. 21.3% of watchers create pull requests after contributing in some other way. Most of these watchers started with submitting code or commenting on pull requests (Table 5).

Table 4. Contributors who Perform a Different Type of Contribution from their First Contribution Type.

Second Type of Contribution	Watchers	Others	Chi-Squared Test
All types	40.1%	16.0%	3276.7**
Report an issue	4.2%	0.7%	921.5***
Submit code	7.7%	3.4%	494.5***
Pull requests	21.3%	4.0%	4783.1**
Comment on Issue	0.6%	5.9%	681.2***
Commit comment	3.5%	1.6%	203.4***
Comment on pull request	2.7%	0.4%	668.9***

Answer to RQ2: Watchers who became contributors are more confident, contribute over a longer period of time, and perform more types of contribution than other contributors.

RQ3: Does a project's programming language impact how its watchers behave?

Watchers are not more likely to become contributors based on the project's programming language. The mean percentage of watchers who become contributors is 5.2% (median 5.0%) when separated by programming language. The mean percentage of contributors who were first watchers is 20.0% (median 20.5%) when divided by programming language.

The type of first contribution varies across different languages. Table 6 shows the top five contribution types by the project's programming language. The projects using C++ receive significantly more comments on commits as a first contribution than the other projects. This could be because of the complexity of the C++ language [10]. Contributors on Go projects are more likely to submit code as their first contribution, while this is very rare as a first contribution on CSS projects which are most likely to receive issues as a first contribution.

Table 7 shows the amount of time watchers wait before contributing, contribution duration and the number of contributions by programming language. While the contribution duration is fairly consistent across programming languages, the length of time watchers wait before contributing and the number of contributions does vary by language. Watchers contribute after

Table 5. Second Type of Contribution based on First Contribution Type.

First Contribution	Second Type of Contribution											
	Watchers that Contribute (%)						Other Contributors (%)					
	RI	SC	PR	CI	CC	CPR	RI	SC	PR	CI	CC	CPR
Report Issue (RI)	-	10.66	7.04	0.50	2.41	0.55	-	2.17	2.23	0.62	0.64	0.15
Submit code (SC)	10.04	-	65.45	1.16	7.44	5.41	1.37	-	12.01	20.17	3.98	0.52
Pull Request (PR)	4.30	4.34	-	0.45	1.96	4.54	2.55	3.53	-	0.76	0.85	4.69
Comment on issue (CI)	1.62	8.44	0.64	-	0.64	0.32	0.15	5.90	0.04	-	0.72	0.004
Comment on commit (CC)	4.27	17.33	6.32	0.08	-	2.46	2.36	9.58	3.49	0.04	-	1.36
Comment on pull requests (CPR)	5.11	32.95	31.81	0.56	6.25	-	7.37	47.5	45.9	0.81	9.01	-

Table 6. First Contribution Type by Programming Language.

	RI	SC	PR	CC	CI
C	26.1%	27.5%	5.9%	0.34%	36.4%
C#	9.6%	46.1%	17.4%	2.2%	24.3%
C++	17.0%	31.9%	4.1%	18.0%	28.5%
CSS	65.5%	4.5%	3.1%	0.2%	26.4%
Go	18.5%	53.5%	8.8%	3.3%	15.1%
Java	26.2%	23.1%	6.2%	8.5%	35.4%
JavaScript	45.1%	16.1%	3.5%	2.2%	32.7%
PHP	8.6%	39.9%	12.0%	3.7%	34.3%
Python	23.2%	35.1%	11.1%	3.1%	26.7%
R	42.4%	15.8%	2.6%	0.5%	38.5%
Ruby	42.3%	19.3%	5.1%	2.4%	30.2%
Scala	22.0%	31.8%	4.0%	2.0%	38.2%
TypeScript	20.1%	29.7%	3.1%	0.4%	42.2%
All	28.5%	27.3%	6.9%	5.0%	31.7%

Table 7. Watcher Contribution by Programming Language.

	Mean Time Before First Contribution (weeks)	Mean Duration of Contribution (weeks)	Mean Contribution Count
C	67.1	15.4	22.9
C#	54.1	19.6	32.8
C++	39.0	16.3	22.0
CSS	36.8	16.2	17.4
Go	39.0	12.6	6.8
Java	47.5	16.7	21.4
JavaScript	44.8	16.8	18.3
PHP	50	16.8	13.0
Python	52.4	16.9	16.3
R	42.9	15.2	9.0
Ruby	74.7	16.9	19.9
Scala	74.4	15.6	10.7
TypeScript	38.5	13.0	26.9
All	55.2	16.4	18.5

an average of only 36.8 weeks on CSS projects, but wait an average of 74.7 weeks on Ruby projects. Projects with longer wait times have more code submitted as a first contribution. In regards to number of contributions, the Go and R languages have an average of 6.8 and 9.0 contributions per watcher. While, C# projects have the largest number of contributions per watcher at 32.8. There is not a clear reason for the differences in contribution counts between programming languages.

Answer to RQ3: Watchers are not more likely to become contributors based on the project's programming language. However, the type of first contribution and the number of contributions received varies across programming languages.

5. CONCLUSION

Contributors who were first watchers account for a significant portion of contributors on GitHub projects. Many GitHub projects do not maintain any formal project documentation, with the exception of a brief wiki and a ReadMe file. Contributors, therefore, learn about a project by reviewing previous comments and developer actions. Watchers have an understanding of the project and its workflow prior to their first contribution due to the notifications they receive. Because of this knowledge, watchers are more confident, contribute over a longer period of time, and perform a greater variety of contribution types. Thus having watchers is important for a project's growth and GitHub was innovative to introduce such a feature. A project's programming language does not impact the amount of watchers it attracts, but it does impact the activity of its contributors.

6. ACKNOWLEDGMENTS

This work was partially funded by the NECSIS grant.

7. REFERENCES

[1] K. Crowston and J. Howison, "The Social Structure of Free and Open Source Software Development," *First Monday*, vol. 10, no. 2, 2005.

[2] G. von Krogh, S. Spaeth, and K.R. Lakhani, "Community, Joining, and Specialization in Open Source Software Innovation: A Case Study," *Research Policy*, vol. 32, no. 7, pp. 1217-1241, July 2003.

[3] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb. "Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository," *Proc. CSCW '12,* ACM, pp. 1277-1286, 2012.

[4] https://github.com/blog/1204-notifications-stars

[5] A. Begel, J. Bosch, and M.-A. Storey, "Social Networking Meets Software Development: Perspectives from GitHub, MSDN, Stack Exchange, and TopCoder," *IEEE Software*, vol. 30, no. 1, pp. 52–66, 2013.

[6] J. Marlow, L. Dabbish, and J. Herbsleb. "Impression Formation in Online Peer Production: Activity Traces and Personal Profiles in GitHub." *Proc. CSCW 13*, pp. 117-128. ACM, 2013.

[7] G. Gousios, "The GHTorent Dataset and Tool Suite." *Proc. MSR 13*, pp. 233-236, IEEE Press, 2013.

[8] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. German, and D. Damian, "The Promises and Perils of Mining GitHub." To appear MSR 14.

[9] R. Pham, L. Singer, O. Liskin, and K. Schneider. "Creating a Shared Understanding of Testing Culture on a Social Coding Site." *Proc. ICSE 13*, pp. 112-121, IEEE, 2013.

[10] R. Hundt, "Loop Recognition in C++/Java/Go/Scala," *Proceedings of Scala Days 2011*, 2011.