On the Usage of Badges in Open Source Packages on GitHub

Damien Legay, Alexandre Decan, Tom Mens
Software Engineering Lab, University of Mons
Mons, Belgium
{damien.legay, alexandre.decan, tom.mens}@umons.ac.be

Abstract

Continuously attracting contributors is key to the health of open source software projects. The appearance of badges in online collaborative development platforms affords maintainers the opportunity to advertise the quality of their project to potential contributors. In this preliminary research, we analyse 14,592 GitHub package repositories for Cargo and 203,029 repositories for Packagist. We measure how prevalent badges are in those repositories, which badges are used, when and how they are introduced, and which combinations of badges co-occur. We find that the most widespread badges convey static information or relay information about the build status of a project. Those badges are typically added early in projects and prior to or at the same time as other badges.

1 Introduction

The rise of distributed collaborative development platforms, such as GitHub, BitBucket and GitLab, allowed thousands of people to remotely work together on the same projects. These platforms provide additional features on top of their underlying version control system to further support distributed collaborative development. Examples of such features are issue tracking, code review, integration with external tools, etc. These features are usually provided through a centralised web-based graphical interface that acts as a portal to showcase a project. Given the variety and quantity of information that can be communicated through such interfaces, it is not surprising that project maintainers have sought a simpler, faster and more concise way to communicate essential information or advertise specific aspects of a software project.

Badges are small images conveying one specific information to the reader at a glance. We found evidence of their use in GitHub dating back to 2011. They typically appear at the top of a project's README file, which is displayed by GitHub on the project repository homepage. Badges can advertise various aspects of a project, e.g., its license license BSD-3-Clause, the code coverage of its test suite coverage 100%, the adopted code style code style creative-area, etc. Some badges act as an incentive to maintain excellence in the qualities they display, lest a bad signal be sent to the project's users and potential contributors. For instance, a code coverage badge creates an incentive to maintain a high code coverage, as otherwise, potential contributors can easily see that the project is poorly tested and, therefore, prone to have hard to detect bugs. Similarly, a badge that measures the quality of the code provides an incentive to maintain a high code quality, to not display on the project's homepage that the code base is of poor quality.

Trockman et al [1] observed a relation between the presence of some badges and specific aspects of the software development process. They found that the presence of dependency management badges correlates with fresher dependencies and the presence of code coverage badges correlates with larger test suites in a project and more tests in pull requests. Contrarily to their expectations, they found that the presence of badges dedicated to offering user support was related to a higher issue resolution time.

As contributors in open source projects are volunteers, they typically have little time to devote to the

In: D. Di Nucci, C. De Roover (eds.): Proceedings of the 18th Belgium-Netherlands Software Evolution Workshop, Brussels, Belgium, 28-11-2019, published at http://ceur-ws.org

projects they contribute to, so they must select such projects with care as they can ill afford to contribute to projects of low quality, which are harder to contribute to and more likely to fail [2–4]. On their part, project integrators need to maintain an influx of contributions in order to keep their projects evolving and growing, as it has long been known that software performing real-world activities must continually grow and continually change to adapt to the environment it evolves in [5]. These laws also apply to open source software [6, 7].

It is known that the number of stars, the time taken to merge pull requests and number of programming languages are the factors most likely to attract contributors [8]. However, little is known about the impact of badges on contributor attraction.

Our overall research goal is to investigate the relation between the presence of badges and the influx of new contributors and new contributions in a project, as well as on the health of projects. It is known, for instance, that the use of continuous integration tools helps catch bugs more efficiently and integrate pull requests faster [9, 10] but not whether badges advertising the use of these tools have any impact on quality. This paper focuses on preliminary but mandatory steps towards this goal by addressing the following research questions. *RQ_0: How prevalent are badges?* This question will help us determine whether the research goal is worthwhile to pursue: if badges are sparsely used, results about the impact of their adoption on contributions may not be statistically significant. Since badges can convey a wide variety of information whose impact on potential contributions may vary, we examine *RQ_1: What are the most frequent badge categories?* As many badges can be used simultaneously in a project, we analyse *RQ_2: How frequent are combinations of badge categories?* The answer to this question will determine whether the effect of badges of one category can be dissociated from those of another category: if two badge categories are always found together and introduced simultaneously, differentiating their impact will be impossible. Finally, we enquire into *RQ_3: When are badges introduced?* This elucidates whether badges are introduced late enough in a project to compare project characteristics prior to and after the adoption of the badge.

2 Methodology

To conduct this study, we need a large dataset of candidate repositories hosted on online collaborative development platforms. As we are interested in studying the effect of badges on contributions, the dataset should exclude git repositories created merely for experimental or personal reasons, or that only show sporadic traces of commit activity [11]. Ideally, it should include a wide range of projects serving different purposes and exhibiting a wide variation in longevity and size.

Online package registries of open source libraries for popular software programming languages constitute good candidates, since they contain a lot of projects, many of them being publicly available on GitHub. We arbitrarily selected two such package registries because we knew from previous work that many of their packages have an associated git repository publicly available on GitHub. These registries are *Cargo* for Rust libraries, and *Packagist* for PHP libraries.

We collected a list of 15,625 packages on Cargo and 216,613 packages on Packagist using their respective official API. We downloaded package metadata for those packages and extracted the link to their associated git repositories. We filtered packages (1) without an associated git repository; (2) whose git repository is no longer available; and (3) corresponding to "spam" packages[1]. Since GitHub hosts nearly all the git repositories of remaining packages ($> 94\%$), and since it is far easier to deal with only a single collaborative development platform, we excluded repositories that were not available on GitHub. The final dataset contains 14,592 package repositories for Cargo and 203,029 package repositories for Packagist.

We cloned all these repositories in July 2019 to extract badge-related historical information. To identify badges, we focused on images contained in the projects' README files. We extracted all images from these README files, taking into account the various supported markup languages (HTML, Markdown and ReST). We manually identified those corresponding to badges following the iterative approach proposed by Trockman et al. [1]. Then, we used git log to analyse the history of these README files to pinpoint the introduction date of each badge.

3 Results

RQ_0: How prevalent are badges?

With this first question, we aim to determine the extent of badge usage in project repositories. We identified for each of the two considered datasets the badges used by their projects. We found 21,884 instances of 109 distinct badges in Cargo projects, and 239,529 instances of 366 distinct badges in Packagist. While there are more badges than projects in both datasets, that does not necessarily imply that all projects use badges. Figure 1 shows the evolution of the number

[1] We manually identified more than 200 packages on Packagist that are not related to software projects, e.g., iphonex-giveaway, captain-marvel-pelicula-completa-uncut, etc. These spam packages are usually quickly removed from Packagist by the maintainers of the registry.

and proportion of projects using at least one badge in Cargo and Packagist. For Cargo, we do not report before 2014 as this only concerns 90 projects out of which only 19 had a badge.

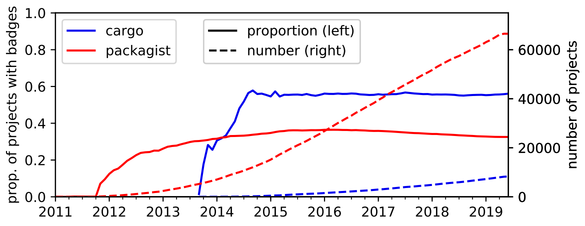

Figure 1: Proportion and number of projects using badges

While there are far more projects with badges in Packagist than in Cargo, we observe a markedly higher badge penetration within Cargo (topping off at 57%) than within Packagist (the highest observed proportion is 36%). For both datasets, the adoption rate eventually reaches a plateau: even though the number of projects using badges keeps increasing more than linearly, it does not supersede the rate of creation of new projects. In both datasets, the most frequent badge is the one reporting the build status of the Travis continuous integration tool (30% and 20% of all badges used, respectively).

RQ_1: What are the most frequent badge categories?

RQ_0 revealed that badges are widely used. However, not all projects use the same badges, and projects may use badges for a variety of different purposes. Furthermore, many badges fulfil a similar role (e.g., several badges can be used to relay the build status of a project, based on different providers such as Travis and Appveyor).

Therefore, we grouped these badges into 7 categories, following the approach of Trockman et al [1]: (1) *build status (Build)* badges signal whether the latest build of a project succeeded or not, e.g., passed all tests build passing; (2) *dependency management (DepMgr)* badges inform about dependency freshness, e.g., whether dependencies are up-to-date or not dependencies up to date; (3) *popularity (Pop)* badges provide characteristics related to the popularity of a project, e.g., number of downloads downloads 5M; (4) *quality assurance (QA)* badges report on aspects related to code quality, e.g., based on the output of some linters code quality 8.26; (5) *support* badges provide links to chats and user forums, e.g., chat on gitter; and (6) *information* (Info) badges communicate various types of information independent of any tool, e.g., the project's license license MIT, version and authors or a link to the project's documentation or website; (7) *other* badges correspond to any badge that does not fit within the previous categories, e.g., donation links beerpay $10.

Table 1: Number (#) and proportion (%) of badge occurrences per badge category for Cargo and Packagist.

		Cargo		Packagist	
category	**first occ.**	#	%	#	%
Info	2014-01-18	11,000	**50%**	69,016	**29%**
Build	2011-11-12	8,130	**37%**	53,478	**22%**
QA	2012-08-31	964	4%	56,167	**23%**
Pop	2013-06-01	606	3%	33,581	**14%**
DepMgr	2013-03-21	534	2%	18,936	8%
Support	2014-01-30	464	2%	3,631	2%
Other	2011-05-24	181	0%	4556	2%

Table 1 reports, for each of these categories, the date of first identification and the number of occurrences and the proportion of badges belonging to each category relative to all of badge occurrences in the dataset. While Cargo projects tend to use more badges than Packagist (on average 1.50 badges per project for Cargo, 1.18 for Packagist), there is less diversity in the badges they use. The starkest contrast is in the usage of *QA* badges, which constitute 23% of the badges found in Packagist projects, but only 4% of the badges in Cargo projects. This is partially explained by the popularity of the Scrutinizer tool in Packagist (18,196 badges are associated with it) which inspects the quality of PHP, Python and Ruby code, but not Rust code. Even other maintainability analysis tools that do support Rust, such as Codeclimate, remain rarely used in Cargo (4 badges found) while they are frequent for Packagist (4,289 badges). The rest of this paper will focus on the categories that account for at least 10% of the badges in at least one of the datasets.

RQ_2: How frequent are combinations of badge categories?

Since a project can make use of several badges at once, this research question aims to quantify co-occurring badges and to identify which combinations of badges are most frequent. Co-occurring badges are clearly not an exception in either dataset: we found that 76% of projects with at least one badge in Cargo have two or more badges at once (77% in Packagist). On average, a project with badges makes use of 2.68 distinct badges in Cargo, and of 3.59 distinct badges in Packagist. If we group badges by category, we have on average 1.94 badge categories in Cargo and 2.89 in Packagist. Badge categories are counted as co-occurring in a project whenever at least one badge of each category is present. Figure 2 shows a Venn diagram of co-occurring badge categories in both datasets.

In Cargo, the most frequent combination by far is the one containing *Build* and *Info* badges,

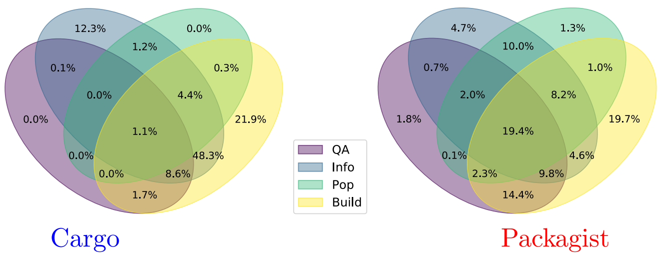

Figure 2: Combinations of badge categories used in Cargo and Packagist projects.

both of them occurring more frequently together (62.4%=48.3+4.4+8.6+1.1) than apart (23.9%=21.9+1.7+0.3 and 13.6%=12.3+1.2+0.1, respectively). We also observe that the lesser-used badges are rarely found alone, they tend to be paired up with a *Build* or an *Info* badge. In Packagist, too, *Build* and *Info* badges are more frequently found together (42%=19.4+9.8+8.2+4.6) than apart (37.4%=19.7+14.4+1+2.3 and 17.4%=10+2+0.7+4.7, respectively). We also observe that nearly one out of five projects with badges in Packagist (19.4%) uses the four considered badge categories at once. In Packagist, badges are less frequently found in isolation. For instance, the proportion of isolated badge categories in Packagist is 27.5% (=19.7+1.3+1.8+4.7) while this proportion reaches 34.2% in Cargo (=21.9+12.3). In both datasets, we found that *Build* is the most frequent isolated badge category by far (21.9% in Cargo, 19.7% in Packagist).

Table 2: Proportion of co-occurring badge categories that were adopted simultaneously in Cargo and Packagist projects.

	QA		**Pop**		**Info**	
Build	54%	38%	64%	71%	65%	71%
QA			51%	49%	50%	43%
Pop					80%	92%

Since we found a non-trivial amount of co-occurring badge categories, in a second step, we examine how frequently badges belonging to different categories were added on the same calendar day in a project. Whenever a category is represented by several badges within a same project, the introduction date of the oldest badge is considered. Table 2 shows the proportion of co-occurring badge categories that were introduced simultaneously. We observe for Cargo that most co-occurring badge categories are adopted simultaneously. For Packagist, it mainly depends on the considered combination. For instance, a large majority of the combinations involving *Build+Pop*, *Build+Info* or *Pop+Info* corresponds to badges introduced on the same day. On the other hand, we observe that 62% (=100-38) of the combinations with *Build+QA* are not adopted simultaneously but in subsequent events.

RQ_3: When are badges introduced?

With RQ_2 we found that most co-occurring badge categories correspond to badge instances introduced on the same day in a project. We now focus on *when* those badges were introduced in a project. For each badge category, we computed the proportion of projects with at least one badge making use of a badge of this category. Figure 3 shows the evolution of these proportions for both datasets. For Cargo, we observe a fast and somewhat massive adoption of *Build* badges: the proportion of projects using such badges went from 2% (September 2013) to 77% (August 2014). A similar observation can be made for *Info* badges to a lesser extent, going from 2% (January 2015) to 59% two years later. The situation is different for Packagist where many projects already existed before badges were available. Indeed, around 5% of projects using badges in Packagist were created before the availability of such badges. This proportion is only 2% for Cargo, which is not surprising given that Rust appeared in 2010. The adoption of badges in Packagist is therefore more gradual, going from practically 0% (June 2013) to 35-40% in two years; with the notable exception of *Build* badges, whose adoption occurred much faster. By August 2012, these badges had been adopted by 60% of the projects, a probable consequence of the introduction of Travis-CI in March 2011.

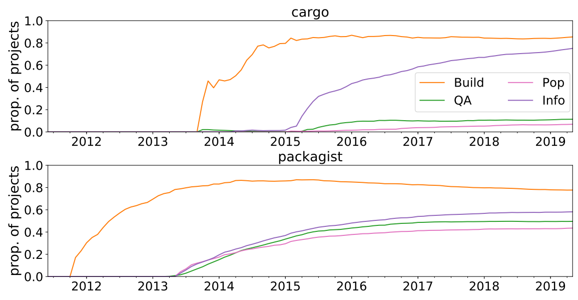

Figure 3: Evolution of the proportion of projects, grouped by badge category.

For each project and badge category, we measured the elapsed time before the first introduction of a badge of a given category in a given project. Since some projects predate the availability of (services advertised by) badges, we measure this time with respect to the *date of the first opportunity* to introduce those badges. So, if the project was created prior to the first occurrence of a category in our dataset, then the date of this first occurrence is used as a baseline. Otherwise, we relied on the creation date of the project as a baseline.

Table 3: Elapsed time before the first instance of a badge category in a project (Cargo and Packagist)

	in days				proportionally			
category	**median**		**mean**		**median**		**mean**	
Info	6	3	119	155	7%	3%	26%	31%
Build	3	2	89	156	3%	2%	21%	30%
QA	10	5	157	163	11%	5%	29%	61%
Pop	4	3	107	170	4%	3%	25%	24%
All categories	6	4	112	166	5%	4%	25%	39%

Table 3 reports on the median and mean of these durations, aggregated by badge category. The left part of the table expresses these durations in days since the date of first opportunity, while the right part expresses them proportionally to the opportunity window (i.e., time between the date of the first opportunity and the last known commit of a project). The huge difference between median and mean values suggests skewed distributions: while a majority of badges are quickly introduced in projects, there are some outliers taking a while to introduce badges. This is especially visible in Packagist: its median values are lower than the ones for Cargo but its mean values are much higher. We also observe that, on average, quality assurance badges were added much later in Cargo projects (median is 10 days vs. 5 for Packagist). In both datasets, *Build* badges are introduced earlier than other badges.

4 Conclusion

We carried out an empirical analysis of the usage of badges in GitHub repositories, with the ultimate goal of determining their impact on contributions to open source projects. As a preliminary step, we sought to determine whether badges were widely used in projects for two popular programming language library registries: Packagist and Cargo.

We found that they are used in more than a third of Packagist projects and more than half of Cargo projects, and that more and more projects tend to use them. Still, badge adoption rates lag behind the rate of appearance of new projects. Then, we categorised badges in seven categories, according to the type of information being relayed by each badge, and measured the relative prevalence of each category. We observed that Packagist projects use a more diverse set of badges than Cargo, the latter mostly sticking to *Build* and *Info* badges. We examined the frequency at which the most common categories co-occurred within the same projects, finding that *Build* badges and *Info* badges are usually found together and that the other categories of badges were rarely found without a corresponding *Build* or *Info* badge. We also found that co-occurring badges were frequently adopted simultaneously. We next examined the temporal aspect of badge adoption and found that the adoption rates of badge categories were either increasing or stable. We also showed that badges were usually added early on, within the first 5% of a project's lifetime, but still a significant amount of projects adopt badges much later. The results we obtained are in line with those of Trockman et al [1] for npm.

As future work, we intend to investigate the impact of badge adoption on contributions. In doing so, an aspect to take into account will be the comparative effort required to maintain some badges over others. We also will quantify the phenomenon of badge removal, determine the reasons why it occurs and what is the impact on contributions.

References

[1] Asher Trockman, Shurui Zhou, Christian Kästner, and Bogdan Vasilescu. Adding sparkle to social coding: An empirical study of repository badges in the npm ecosystem. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, pages 511–522, New York, NY, USA, 2018. ACM.

[2] D. Riehle, P. Riemer, C. Kolassa, and M. Schmidt. Paid vs. volunteer work in open source. In *2014 47th Hawaii International Conference on System Sciences*, pages 3286–3295, Jan 2014.

[3] Israr Qureshi and Yulin Fang. Socialization in open source software projects: A growth mixture modeling approach. *Organizational Research Methods*, 14(1):208–238, 2011.

[4] Jailton Coelho and Marco Tulio Valente. Why modern open source projects fail. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 186–196. ACM, 2017.

[5] M. M. Lehman, J. F. Ramil, P. D. Wernick, D. E. Perry, and W. M. Turski. Metrics and laws of software evolution-the nineties view. In *Proceedings Fourth International Software Metrics Symposium*, pages 20–32, Nov 1997.

[6] Taranjeet Kaur, Nisha Ratti, and Parminder Kaur. Applicability of Lehman laws on open source evolution: a case study. *International Journal of Computer Applications*, 93(18):0975–8887, 2014.

[7] Godfrey and Qiang Tu. Evolution in open source software: a case study. In *Proceedings 2000 International Conference on Software Maintenance*, pages 131–142, Oct 2000.

[8] Felipe Fronchetti, Igor Wiese, Gustavo Pinto, and Igor Steinmacher. What attracts newcomers to onboard on OSS projects? tl;dr: Popularity. In Francis Bordeleau, Alberto Sillitti, Paulo Meirelles, and Valentina Lenarduzzi, editors, *Open Source Systems*, pages 91–103. Springer, 2019.

[9] Michael Hilton, Timothy Tunnell, Kai Huang, Darko Marinov, and Danny Dig. Usage, costs, and benefits of continuous integration in open-source projects. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, pages 426–437. ACM, 2016.

[10] Bogdan Vasilescu, Yue Yu, Huaimin Wang, Premkumar Devanbu, and Vladimir Filkov. Quality and productivity outcomes relating to continuous integration in github. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pages 805–816. ACM, 2015.

[11] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. The promises and perils of mining GitHub. In *Working Conference on Mining Software Repositories (MSR)*, pages 92–101, New York, NY, USA, 2014. ACM.