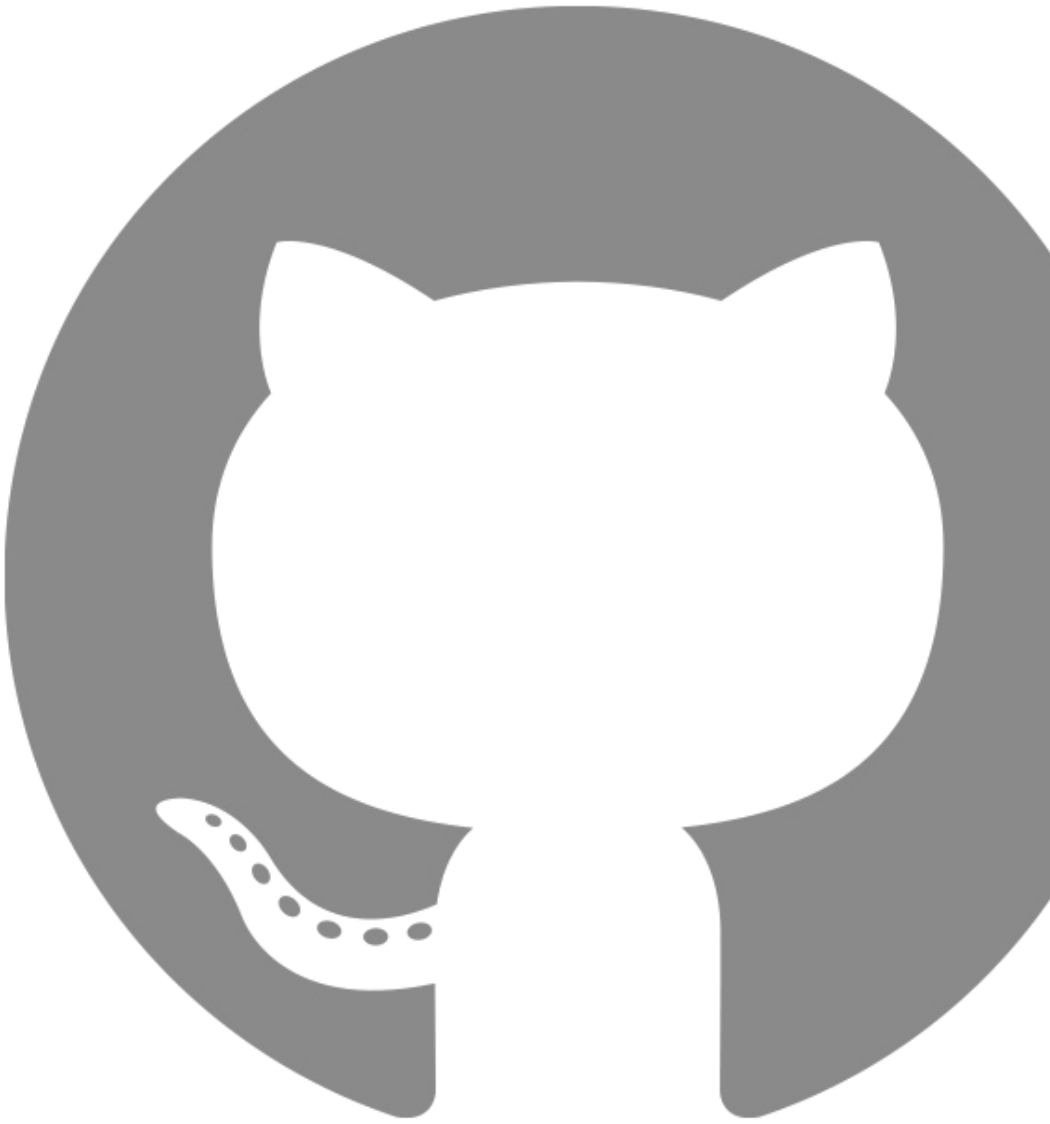
Submission date : 12/01/2015

The Use of GitHub by Software Development Students

Human Centered Informatics - 7th Semester - ICT based data collection and analysis - Assignment 1



Guillaume Salmon

21397 characters - 9 standard pages

Introduction

The Git protocol is nowadays the most used versioning system used in software development. Its key feature which differentiates it from previous versioning tools (like SVN or CVS) is a totally decentralized architecture. Git can work locally on a computer and allow local versioning without any external repository. Though it is most often used paired with a remote repository compatible with the protocol on which the user can upload the code he is developing, upload modifications as he is developing them and allow collaborators to do the same.

Amongst the repository hosting services compatible with Git, GitHub is, with 87.5% of Git users using it, the largest[1] despite being around for only 7 years. The possible reasons for this success are its ease of use and its affordability, with even the possibility to have a free account including most of its functionalities.

If in a professional environment other services are also used, especially self hosted services which allow private repositories to be hosted inexpensively on a preexistent infrastructure. Though GitHub is not only used as a hosting service for code. They also offer a powerful issue system which allows involved users and testers to give feedback on bugs, missing functionalities or desired improvements directly to the developers who can even use rely on them to organize the development process. This way to deal with users' feedback from within the development process is similar to agile methods and another GitHub's functionality also goes in this direction, the milestones. Milestones are a list of prioritized issues to fix and functionalities to add that are used as software versions by developers, helps planning deadlines within the process and helps integrating some project management aspect at a low level. It is especially useful on open-source projects when teams are geographically scattered, and for developing teams without dedicated project management resources. Previous research has indeed shown that long distance collaboration tends to be common amongst GitHub users while projects involving a low amount of collaborators (and thus less resources for

[1] https://git.wiki.kernel.org/index.php/GitSurvey2011#19._Which_git_hosting_site.28s.29_do_you_use_for_your_project.28s.29.3F

management) tends to have them more concentrated geographically (Lima, Rossi & Musolesi, 2014). Pull-requests are also a central feature from GitHub. They allow users who work on a copy of an existing project to request the integration of a part of their code in the original project. This feature is used on open-source projects to share features amongst forked projects (a fork is a copy on an independent repository of a project).

Although this is a recent trend, nowadays as a student in software development at the university, one is extremely likely to be asked to use GitHub to host a development project (Zagalsky, Feliciano, Storey, Zhao & Wang, 2015). GitHub, Inc. is aware of it and they promote[2] actively the use of their platform by universities and university students. But if on the scale of industrial projects or widely spread open source projects the use of remote repositories makes a lot of sense, at a universitary scale, does students actually use all the features which makes GitHub such a must ? A universitary project is usually developed within a few months by a limited amount of collaborators, most of the time abandoned afterwards by its developers, and as an educational project, it is rare that it actually covers functionalities which have not already been covered by more serious open source projects, which makes its availability to anyone else not especially useful. In addition the strong regulations about plagiarism which apply most of the time in university contexts is not especially compatible with the code sharing and open-source philosophy behind GitHub.

I will thus make an inquiry on whether students truly exploit the main tools offered by GitHub. A number of features in GitHub remind of social network, but studying those aspects would broaden the field of this study to the social aspects of development in open source communities. These aspects have been studied extensively by some researchers (Lima et al., 2014), but are too wide to be studied thoroughly here. I will focus here on the use of the tools offered by GitHub directly related to the development and the life of a project which are mainly the issue system, milestones system and the pull request system.

[2] https://education.github.com/

Methods

Survey

In order to study the user practice of students with GitHub, I have run a survey amongst master and bachelor software development students mainly in the Toulouse University III[3] in France, but also from other universities and IT schools (ENSEEIHT[4], Buenos Aires University[5], Toulouse University II[6]).

The survey was kept small, limiting the types of data collected, in order to make people more likely to get through and get more answers. Multiple choice questions were prefered to make the data easier to handle.

The choice of a survey was lead by several concerns. The main issue while making this inquiry was a clear lack of time. While it is easy for me to get contacts amongst software development students in my home city, as a fresh resident in Denmark, it would have taken more time to get a sufficient panel of software development students here in Aalborg for interviews or ethnography. And those methods, even if I had found a consequent panel of students, would have been too heavy to carry out on the span of one week. The second reason is that a survey allows me to get quantitative data that matches the data I can get through the GitHub API allowing a direct comparison between the two sets of data.

The survey was conducted through an online questionnaire, made in Google forms, since it is a tool that offers the functionalities needed. The styling is premade, and the interaction design is already in place. This makes it easily accessible, both for us as researchers and for the respondents. It is also fair to assume that software development students would be comfortable with filling an online questionnaire. Even though most of the respondents were French I still formulated the questionnaire in English. The main reason is that I shared the questionnaire via Facebook, aiming primarily the students from certain masters and bachelors in my home city but leaving it available for any software development student with some knowledge about GitHub, and I anticipated

[3] http://www.univ-tlse3.fr/
[4] http://www.enseeiht.fr/en/index.html
[5] http://www.uba.ar/ingles/index03.php
[6] http://www.univ-tlse2.fr/home/

having answers also from some international students, which indeed have been the case.

In order to design the questionnaire, no real risk of contextual effect was to be wary of since the questions were addressing the use of precise tools and thus could hardly be misleading. A good practice when designing a questionnaire is to start with easy questions, and increase the complexity of questions progressively. In the case of this questionnaire, the difficulty of questions is relatively stable, though I took care of putting the more general questions at the beginning, then addressing the more precise questions about the tools to finally end with the open question (Spector, 2013). The whole questionnaire is available in annex.

The two first questions are there to give an idea of the respondents' level of familiarity with GitHub. The amount of projects they are hosting and their tendency to use it or not for their projects give a good indicator on whether they are likely to exploit the possibilities offered by the platform whenever they use it. Then comes the questions about specific tools. I asked about the students usage of README files, issues, pull requests and milestones. The proposed answers allows me to know whether the respondents know about the feature, how often they use it, if it is only when asked to in a university project and for milestones and issues if it helps them to manage the development process. I also asked a more personal question on whether or not they think their GitHub account should be used to assess their coding abilities by recruiters. This question might seem off topic, but this is nowadays a common practice, so it is relevant to understand the way students use their GitHub account to know if they consider it as solely a personal tool or also as a public display of their abilities. Finally the last question is about the other Git based code hosting systems students might use and if they use them more or less intensively than GitHub. This also might seem a bit off topic, but it is important to know it because it might influ on their user practice on GitHub. The survey got 37 respondents over the span of two days, which, while not being a lot, is still a bit more than one third of the amount of accounts studied to get the log data, making the two samples of the same order of magnitude. The answers to the questionnaire are available as figures in the Analysis section of this paper and the answers to the last question are detailed in annex.

Logdata

In order to have some reference data, I used some log data from GitHub. I accessed this data through the GitHub API. Its documentation is available at https://developer.github.com/v3/ . The goal is to have some reference data about the general users. I accessed the data of the 100 first users returned by the API when asked to list GitHub users. To use this API, I had to write a script in bash[7] which is the default shell used in command line interfaces under the main Linux distributions and under MacOS X. The script is available in annex, requires a GitHub login and password, up to 50 minutes to run, the curl library installed, runs on a recent Ubuntu or Debian and although not tested should run as well on MacOS X. The amount of requests to the API allowed per hour being of 5000 for an authentified user, it is impossible to run the script twice within an hour (the script does about 4700 requests per run).

To understand the quality of the collected data, some background knowledge about the studied GitHub tools is necessary. The collected log data consists in the average amount of repositories per user, the average minimum amount of issues per repository, the average size of a repository, the minimum amount of users using milestones and the amount of user using pull-requests. All this data is available for any public repository. It thus consists only of open-source projects, which means that it is mainly non-professional projects, but rather personal and community based projects.

Counting the repositories is trivial, the information is directly available for each user through an URL with https://api.github.com/users/[username] as format, in the "public_repos" category.

Knowing how much of them use pull-requests is possible by accessing the list of repositories for each user through an url with https://api.github.com/users/[username]/repos as format, then for each repository accessing the list of the formulated pull-requests through the url https://api.github.com/repos/[username]/[reponame]/pulls. Then if the answer from the API is bigger than 138 bytes, it means that it contains pull-requests while below or equal

[7] http://www.gnu.org/software/bash/

to 138 bytes means it contains nothing, an empty set or information stating that milestones are disabled for the repository.

Retrieving the amount of issues per repository is a bit more tricky. The detailed data about all repositories from a user is available through the url https://api.github.com/users/[username]/repos but it gives only the amount of open issues. The principle behind issues is that they are raised in an “open” state when a user or a tester finds a problem, which means that they are still unsolved, and a number is assigned to them. When a developer fixes the problem, he closes the issue. The number assigned to a new issue corresponds to the sum of all previously raised (open or closed) issues plus one. This means that accessing only open issues does not give a real idea on the total amount of issues. But through an url with the following format https://api.github.com/repos/[username]/[reponame]/issues it is possible to access the details of all open issues from a repository and thus access the numbers assigned to theses issues. So using the highest number assigned to a still open issue for this repository, it is possible to know the minimum amount of other issues, open or closed. So by adding for all repositories the number of the most recent open issue, then dividing this total amount by the amount of repositories processed, we have an average amount of issues per repository. This number probably does not reflect the real average amount of issues per repository, but we thus know for sure that the real average amount of issue per repository can only be bigger.

I used a similar method to count milestones. Milestones are a list of issues which are required to be fixed and are used to define intermediate versions of a software under development. Only non completed milestones can be accessed through the API and they are numbered in the same way as issues. They are accessible for each repository through the url https://api.github.com/repos/[username]/[reponame]/[milestones]. By looking at the size of the answer from the API, it is possible to count the users who have currently open milestones. The total amount of user who use milestones can only be equal or bigger than this.

The nature of the collected data is comparable to what was collected amongst the software student through the survey.

After running the script, the collected data is the following :

On 100 users 49 have used pull requests, at least 52 have used milestones and there is on average 60 repositories per user. The data from 2374 of these repositories have been analyzed. This is only a part of all these users' repositories because not all the repositories' data is available through the GitHub API. On average, a minimum of 2 issues have been found for each repository and each repository weights on average 6139 bytes. Taking in consideration only the repositories that we are sure to be containing issues, on average a minimum of 27 issues have been found for each repository.

Analysis and discussion

First it is important to note that the survey confirms the status of GitHub as most people's favourite Git based code hosting system. Out of the thirty seven respondents twenty one are using only GitHub, and amongst the sixteen others only one claims using another system more than GitHub (BitBucket, because of the possibility to have private repositories without paying anything). But we can also see from the answers that several students use either GitLab (another Git based code hosting system), closed source repositories either on BitBucket or GitLab, or even no server at all, using only the local versioning system from Git for some of their projects. This shows that the community aspects are not necessarily a primary concern for them.

Though they still tend to use the development oriented features at a similar level as the average user. Slightly less than half the students declare using milestones at least sometimes and not only when asked to in a university project (Fig. 1), while according to the log data, at least fifty two from one hundred users used milestones. Students still use them less than the average user but the difference does not appear as important. The same constatation goes for issues. We know for sure that since the milestone system uses issues, at least fifty two out of one hundred users use issues according to the log data. From the survey we know that 54% of students are using them at least sometimes also when not explicitly asked to do so (Fig. 2). Concerning the pull requests, 49% of users are using them according to the log data while according to the survey 67.5% of students claim using it at least sometimes and 35% are even using it often (Fig.3).

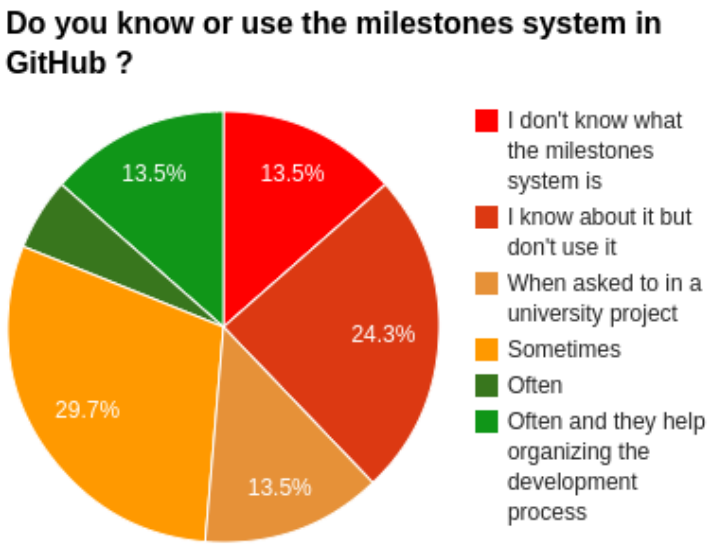


Figure 1

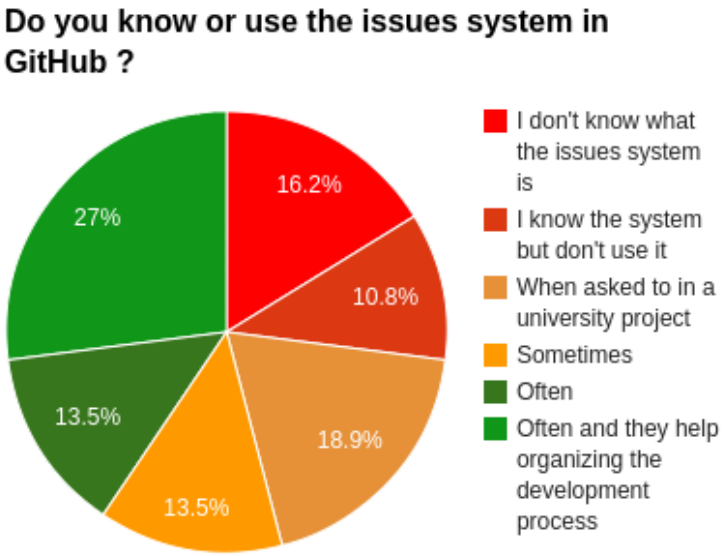


Figure 2

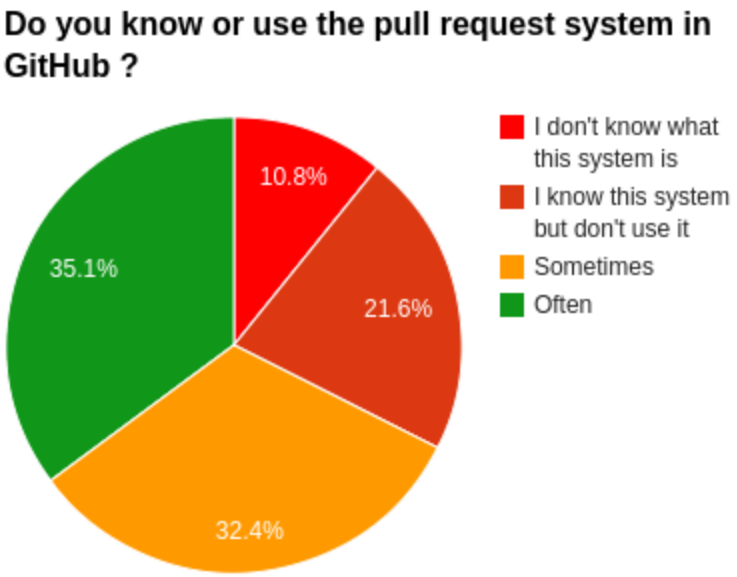


Figure 3

These are impressive results since pull requests are not especially useful in a university context. Plagiarism is usually heavily discouraged and using pull requests implies either that some code in the project comes from another project or that another project has copied the project's code. The important use of milestones is less astonishing given that university projects are usually in small groups and thus the project management has to be dealt by everyone involved in the project. Though it still makes more sense to use them for projects with a certain lifespan, while university project usually have a lifespan of a few month maximum.

It is also interesting to note that from the log data, we know that for projects that we know to have issues, there is at least 27 issues in average, while for projects in general, this minimum averages drops to 2. It shows that issues are used a lot on some projects but seem rather inexistant on most of them. Though we know also that at least 52% of users are sometimes using issues. We can thus infer that there is probably a small amount of key projects for each user on which this type of tools are really useful. But this supposition cannot be applied to software development students. We know from the survey that 97.2% of them participated to less than 30 project, with 48.6% of them

having participated to less than 5 projects (Fig. 4). As a comparison, from the log data, we know that GitHub users have on average 60 repositories hosted by the service. Using so intensively issues, milestones and pull requests with such a small amount of projects of would mean either that students host only key projects on their account or that they use those tools more than necessary. But the data collected through the survey indicates that more than half of the surveyed students are actually usually hosting their development projects on GitHub (Fig. 5), so we can deduce that they probably actually use the tools offered by GitHub more than necessary.

A simple explanation to this phenomenon would be that as students learning to use new tools, it is normal that they tend to use them more than necessary in order to get used to them. I would have an objection though. The amount of GitHub users is growing extremely fast and the amount of repositories is currently increasing exponentially. On GitHub, most of users are new users or at least recent users. To quote GitHub, Inc.'s blog, "In fact, over 5.5M repositories — more than half of the repositories on the site — were created this year alone."[8]. So it is not only the students who are currently learning how to use the tools, the phenomenon should be visible on the log data from random users.

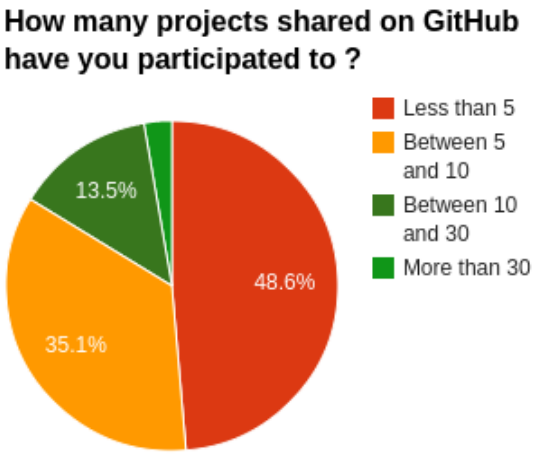


Figure 4

[8] https://github.com/blog/1724-10-million-repositories

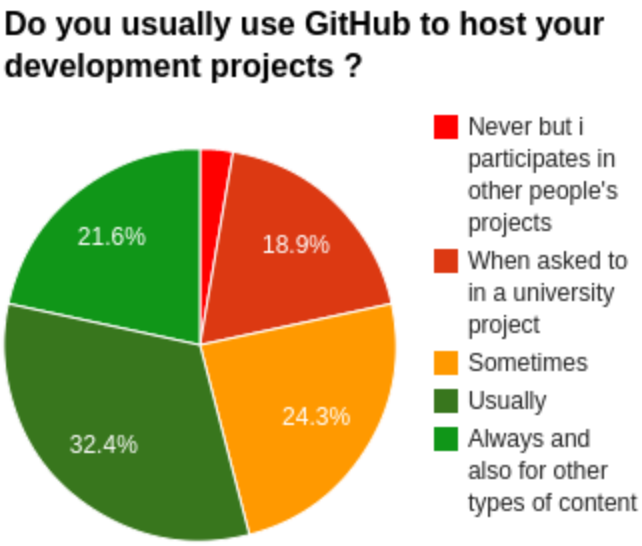


Figure 5

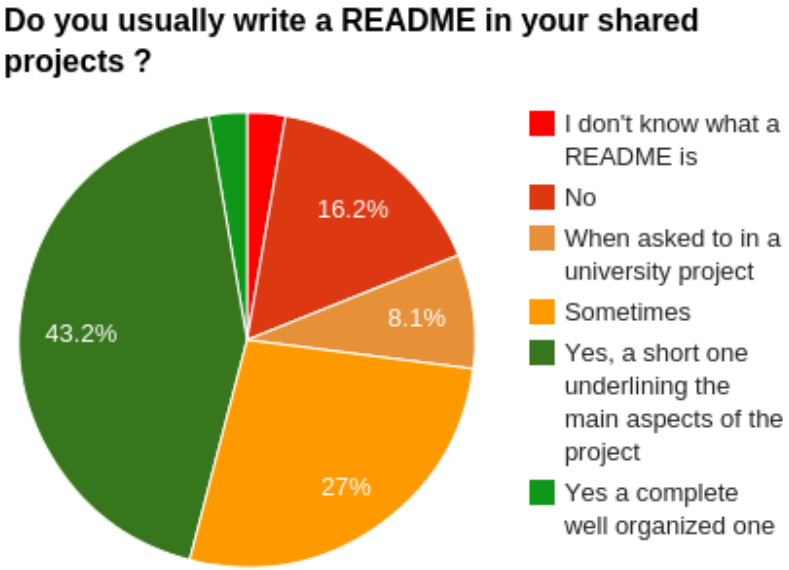


Figure 6

Another possibility is that the use students have of GitHub is different from the use the major part of the GitHub community have of it. We saw that the open-source community

aspect did not seem to have a big importance for students, which is also visible in the questionnaire's results since a really small amount of students actually write a complete README file in their projects (Fig. 6), while this is a must for any open source project if the developers want any new collaborators to be able to join. Some of them tend to use also other service which allow private repositories, and university projects are generally not worth spending much time on using an issue system which aims primarily at allowing users to give feedback and request features since there is usually not really any final users. In addition it is rare that students keep maintaining their projects once the semester is finished. But the students primary aim is often simply to be employable once their studies finished. Since it has become a usual thing for recruiters to look at candidates' GitHub accounts to assess their abilities, it is fairly reasonable to assume that students tend to use the tools offered by GitHub more than necessary simply because GitHub accounts are being used as CVs both by employers and candidates. In the survey we can see that more than half the students consider it as relevant for recruiters to use their account to assess their coding abilities (Fig 7). If an account is used as a showcase for one's abilities to take part of a development project, it is normal to use the tools regularly in order to show a certain proficiency. It would require further investigations to validate or invalidate this assumption, however it seems likely to have an impact on the way software development students use GitHub.

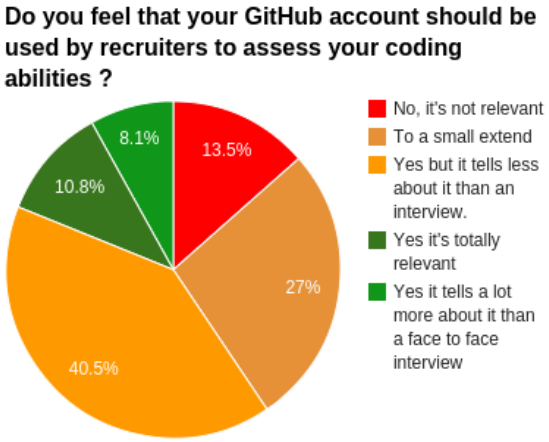


Figure 7

Limitations

This study suffers from several limitations. On the literature side, GitHub being a relatively new system, it is still difficult to find relevant literature on the subject.
About the survey, if I had more time for this inquiry I would have spread the questionnaire more widely. Most of the answers I got come from students from extremely similar studies and almost all of them are located in France. Given the important variations between the different educational systems in various countries, it would have been more appropriate to have results from a panel of students coming from different countries and different systems.
Finally the main limitations came from the GitHub API which did not allow me to access all the kinds of data I wanted to. In addition, the important amount of requests it sends to GitHub's servers makes it long to run. With about one hour to run it entirely, it makes the testing process especially long and difficult and thus slowed down the writing of the script considerably, which left less time than I expected to analyze its results.

Reference list

Lima, A., Rossi, L., & Musolesi, M. (2014, May). Coding Together at Scale: GitHub as a Collaborative Social Network. In Eighth International AAAI Conference on Weblogs and Social Media.

Zagalsky, A., Feliciano, J., Storey, M. A., Zhao, Y., & Wang, W. (2015). The Emergence of GitHub as a Collaborative Platform for Education.

Spector, P. (2013). Survey Design and Measure Development. In *The Oxford Handbook of Quantitative Methods in Psychology*, Vol. 1. T. D. Little

Annexes

Script

```
#!/bin/bash
# This script is under the GPL Licence

date -u

# Gets a list of users from the github API
read -p "Enter your GitHub username : " username
read -p "Enter the password for $username : " -s pass
echo ""
echo "Retrieving usernames..."
curl -u $username:$pass https://api.github.com/users > github_users

# Initiates folders, files and variables
touch userlist
echo "" > userlist
mkdir users &> /dev/null
touch userdetails
echo "" > userdetails
count=0
countRep=0
activeRep=0
nrepos=0
nissues=0
pr_users=0
m_users=0


# For each user, retrieves only its username
while read p
do
  if [[ "$p" == *"\"login\""* ]]
  then
        echo $p | cut -d '"' -f 4 >> userlist
  fi
done < github_users
echo "Done retrieving usernames, retrieving user details... (can be long)"
```

```
# For each username, retrieves the corresponding details
while read p
do
        if [ "$p" != "" ]
        then
                let "count += 1"
                echo "$count/100"
                echo https://api.github.com/users/$p
                curl -u $username:$pass https://api.github.com/users/$p >> userdetails
                curl -u $username:$pass https://api.github.com/users/$p/repos > $p
                mv "$p" "users/$p" &> /dev/null
        fi
done < userlist
sed -i 's/,/ ,/g' userdetails &> /dev/null
echo "Done retrieving user details"

# Get the data from the retrieved details
echo "Retrieving and crushing repositories' data from all users... (can be long enough to take
a coffee)"
while read p
do
        if [[ "$p" == *"\"public_repos\""* ]]
        then
                rep=`echo $p | cut -d " " -f 2 `
                let "nrepos += rep"
        fi
done < userdetails
let "nrepos /= count"
cp userlist users/userlist
cd users
let "count = 0"
tsize=0
let "nOpIssues = 0"
while read u
do
        # For current user
        if [ "$u" != "" ]
        then
                does_pr=0
                uses_m=0
                let "count += 1"
```

```
		echo "User $count of 100 being processed"
		repo_name=""
		sed -i 's/,/ ,/g' $u &> /dev/null
		while read l
		do
			# For current repository of current user
			if [[ "$l" == *"\"name\""* ]]
			then
				let "countRep += 1"
				repo_name=`echo $l | cut -d '"' -f 4`
				touch $repo_name

				milestones="_milestones"
				touch "$repo_name$milestones"
				fsize=`stat -c%s "$repo_name$milestones"`
				# Hack which allows upon several launches to solve the API limit
problem if encountered
				if [ $fsize -eq 0 ] || [ $fsize -gt 119 ]
				then
					curl -u $username:$pass
https://api.github.com/repos/$u/$repo_name/milestones > "$repo_name$milestones"
					foo=0
				fi
				fsize=`stat -c%s "$repo_name$milestones"`
				if [ $fsizeP -gt 138 ]
				then
					uses_m=1
				fi

				pr="_pullrequest"
				touch "$repo_name$pr"
				fsizeP=`stat -c%s "$repo_name$pr"`
				# Hack which allows upon several launches to solve the API limit
problem if encountered
				if [ $fsizeP -eq 0 ] || [ $fsizeP -gt 119 ]
				then
					curl -u $username:$pass
https://api.github.com/repos/$u/$repo_name/pulls > "$repo_name$pr"
					foo=0
				fi
				fsizeP=`stat -c%s "$repo_name$pr"`
```

```
                        if [ $fsizeP -gt 138 ]
                        then
                                does_pr=1
                        fi

                fi
                if [[ "$l" == *"\"open_issues_count\""* ]]
                then
                        issue=`echo $l | cut -d " " -f 2`
                        max_issue=0
                        if [ $issue -ne 0 ]
                        then
                                let "activeRep += 1"
                                curl                    -u                  $username:$pass
https://api.github.com/repos/$u/$repo_name/issues >> $repo_name
                                sed -i 's/,/ ,/g' $repo_name &> /dev/null
                                while read i
                                #  For  current  open  issue  from  current  repository  of
current user
                                # Retrieve the minimum number of issues on the repository
based on the highest issue number from open issues
                                do
                                        if [[ "$i" == *"\"number\""* ]]
                                        then
                                                cur_issue=`echo $i | cut -d " " -f 2`
                                                if [ $cur_issue -gt $max_issue ]
                                                then
                                                        let "max_issue = cur_issue"
                                                fi
                                        fi
                                done < "$repo_name"
                                let "nOpIssues += max_issue"
                        fi
                        let "nissues += max_issue"
                fi
                if [[ "$l" == *"\"size\""* ]]
                then
                        size=`echo $l | cut -d " " -f 2`
                        let "tsize += size"
                fi
        done < "$u"
```

```
                if [ $does_pr -eq 1 ]
                then
                        let "pr_users += 1"
                fi
                if [ $uses_m -eq 1 ]
                then
                        let "m_users += 1"
                fi
        fi
done < userlist

let "nissues /= countRep"
let "nOpIssues /= activeRep"
let "tsize /= countRep"
echo " "
echo "On $count users $pr_users have used pull requests, $m_users have used milestones and there is on average $nrepos repositories per user"
echo "The data from $countRep of these repositories have been analyzed. "
echo "Only a part of all these user's repositories have been analyzed because not all the repositories' data is available through the GitHub API"
echo "On average, a minimum of $nissues issues have been found for each repository and each repository weights on average $tsize bytes"
echo "Taking in consideration only the repositories that we are sure to be currently active, on average a minimum of $nOpIssues issues have been found for each repository"

date -u
echo " "
exit
```

Questionnaire

Do you usually use GitHub to host your development projects ?

- Never but I participate in other people's projects
- When asked to in a university project
- Sometimes
- Usually
- Always and also for other types of content

How many projects shared on GitHub have you participated to ?

If you have a doubt about this question, go to https://github.com/[YOUR_USERNAME]?tab=repositories and take a quick look to the list of your projects there. I promise, that's the last time you have to count something.

- Less than 5
- Between 5 and 10
- Between 10 and 30
- More than 30

Do you usually write a README in your shared projects ?

- I don't know what a README is
- No
- When asked to in a university project
- Sometimes
- Yes, a short one underlining the main aspects of the project
- Yes, a complete well organized one

Do you know or use the issues system in GitHub ?

- I don't know what the issues system is
- I know the system but don't use it
- When asked to in a university project
- Sometimes
- Often
- Often and they help organizing the development process

Do you know or use the pull request system in GitHub ?

- I don't know what this system is
- I know this system but don't use it
- When asked to in a university project
- Sometimes
- Often

Do you know or use the milestones system in GitHub ?

- I don't know what the milestones system is
- I know about it but don't use it
- When asked to in a university project
- Sometimes
- Often
- Often and they help organizing the development process

Do you feel that your GitHub account should be used by recruiters to assess your coding abilities ?

- No, it's not relevant
- To a small extend
- Yes but it tells less about it than an interview.
- Yes it's totally relevant
- Yes it tells a lot more about it than a face to face interview

Do you use other Git based repositories systems than GitHub ? Please precise if you use them more or less than GitHub.

Answers to the questionnaire's last entry

21 respondents answered using only GitHub. For the 16 remaining respondents, I got the following answers :

bitbucket.org, less used than github

I use most of the time my Git repo on my home hosted server.

I only use git alone when i dont use github. because it allows me to have a trace of what i did.

I like bitbucket more, because it offers private free repository with no conditions..

Gitlab for self-hosting, less than Github

Only in the enterprise where I work (alternance) : gitlab. But for universities and personnal projects I only use Github.

Bitbucket (10 times less than Github)

Bitbucket, used less than GitHub. Prefer GitHub.

Bitbucket, use it less than github

I use a GitLab server installed on my dedicated server, it has the same features than github, but allows closed source (closed source on github is not free). Same use as github

bitbucket, gitlab - less

bitbucket as the same

bitbucket, gitlab, for private repos only

SourceTree less used