Software Analytics

Is The Quality Of A Project Connected ToThe Tests To Which It Was Submitted ?

Kim Leng Chhun
HES-SO
kimleng.chhun@master.hes-so.ch

Carol Hubert
HES-SO
carol.hubert@master.hes-so.ch

Manel Hajri Ep Mrad
HES-SO
manel.hajriepmrad@master.hes-so.ch

Jérôme Schroeter
HES-SO
jerome.schroeter@master.hes-so.ch

***Abstract*—This article explains our researches on what makes a good project, and if these criteria are somewhat connected or not to the tests that were ran on this project.**

Keywords
Software Analytics, Github, Travis Torrent, Test Quality, Software Evolution

I. Introduction

Our first question was to determine some important criteria allowing to know whether a project may be considered as a good one, or not. The second question was to know what kind of tests were run on a project, how many of them and how often they were run. Last, but not least : are these two questions somehow related ? Do the tests make a good project ?

We can find these data in two different sources, the Github and Travis Torrent databases.

II. Approach

A. Identify the criteria that make a good project

In a first time, we tried to identify the criteria that allow us to determine whether a project is good or not. In the Github database, we picked up the following data :

- Number of stars
- Number of followers/subscribers
- Number of forks

B. Identify the tests run in a project, and other criteria

We then wanted to know if some tests had been run on the considered project, how many of them passed or failed. We recorded as well the size of the project, that is how many people worked on it, and how many builds there were.

C. Identify the relationship between the criteria of a good project, the tests run in the project and the other criteria

We finally tried to see whether these data could be connected somehow, whether a good project has many successful tests and builds, or whether a project with no tests can be good. For this part we used the Tableau software, that allowed us to see the data in the form of clearly comprehensive graphics.

III. Sources

A. Github

We used a javascript script[1] to extract the desired data from the Github Api. We used Docker to run the script, in the following way :

1°) Definition of a docker file

This allows to mount the virtual machine environment of Docker. We then install NodeJS and the dependency packages, and copy the script folder from the host to the docker machine. Finally we launch the script (cmd command).

2°) Definition of docker-compose.yml

Though we have only one service to mount, we do use docker-compose to make the run easier, and get some more explicit logs. We specify the image (*docker file*) to run, and the port connexion (*8080*).

We tried to configure our project for sharing folders between docker machine and host with the volume command in docker-compose, but we failed. Running the script generated an error message that we couldn't resolve.

[1] getTravisGithubData.js

B. *Travis Torrent*

We downloaded this database in its CSV format[2], so we could open it directly in our Tableau software and get immediately the data that interested us.

IV. Results

The basic idea was to analyze a whole lot of projects, but we first started with a ‘good’ project, in order to see its characteristics. This project has many stars and forks, and very few open issues, though there aren’t many subscribers. It is named ‘*activeadmin’*.

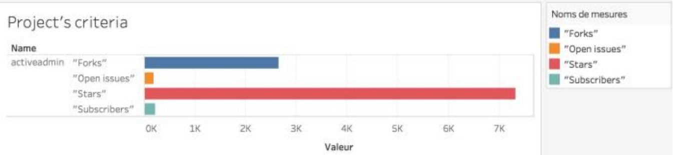

Figure 1 – good project

We tried to see which were the interesting characteristics in this project. Our objective was to select the most interesting criteria, in order to apply them and analyze all the projects.

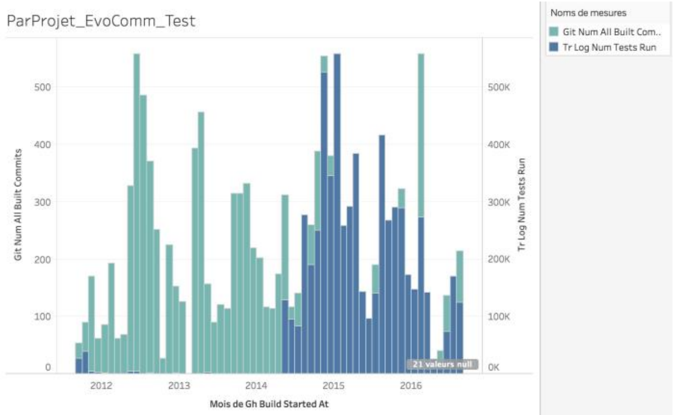

Figure 2 - builds and tests

We observe that many builds and tests were run on this project. We then wanted see whether these tests were successful or not. The next two pictures show us that. In the first picture, ‘faux’ shows when the build status is failed, and ‘vrai’ when the build status is succeeded. We can see that there are more failed tests when the builds are failed as well. On the contrary, when the builds are successful, there are very few failed tests.

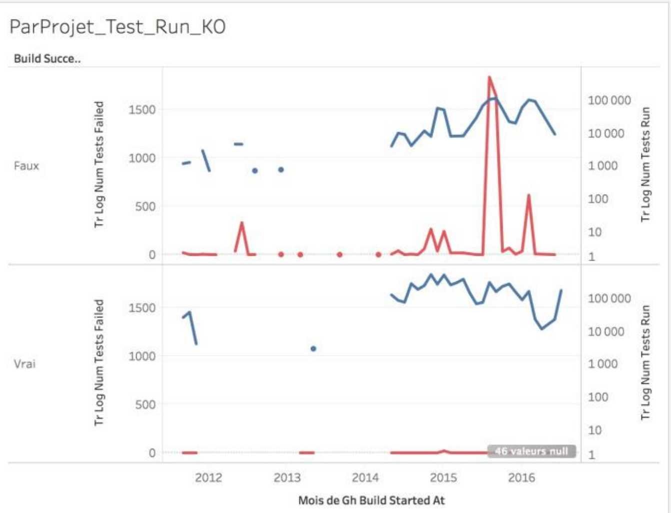

Figure 3 – successful/failed tests by successful build

In the second picture, ‘faux’ shows the tests for external members, while ‘true’ shows the tests of the core team. We cannot see any important difference in the results of these two groups. That’s why we didn’t make this separation for our further analyzes.

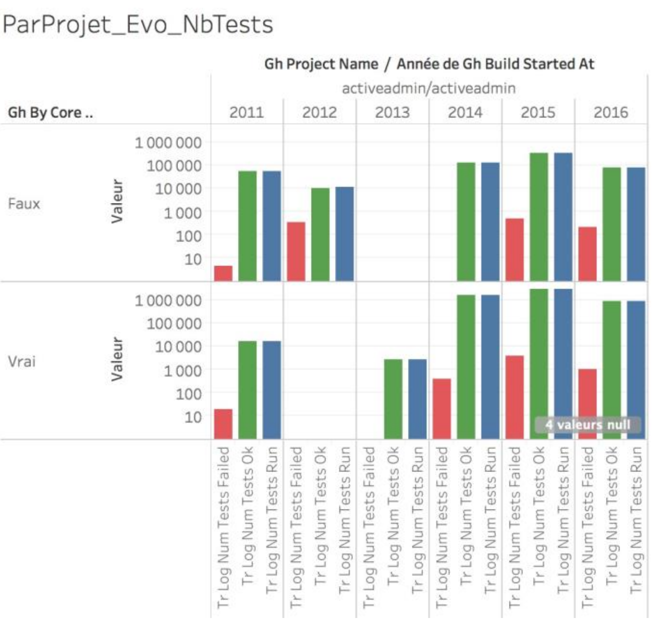

Figure 4 – successful/failed tests by team

The fifth picture shows the evolution of the team size along the years. We can see that it’s very inconstant, starting with only a few members, ending with quite a lot, and very different values in between.

[2] travistorrent_8_2_2017.csv

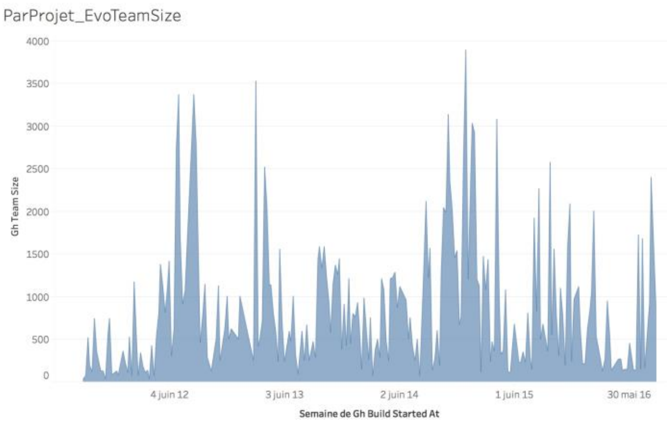

Figure 5 - team size

The other characteristics we could find didn't show any interesting results, so we didn't use them.

We then looked for data concerning all the projects in the Travis Torrent file. We divided our projects in two categories : if the numbers of forks, of stars and of subscribers are all over 1'000, it's a good project, else it's a bad one. For each category, we analyzed the number of built commits, failed tests and total tests.

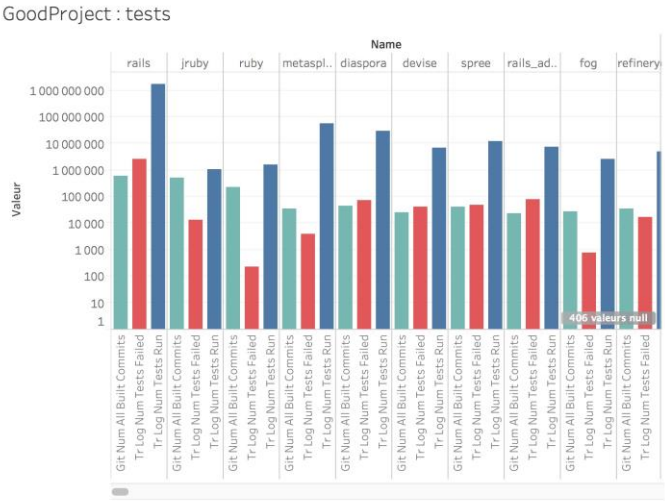

Figure 6 - good projects

We can see that our good projects usually have many tests, even more than one billion for one of them, about 10'000'000 in average.

As for the bad projects, the number of tests is far below, with a maximum under 10'000'000, and even projects with no tests at all.

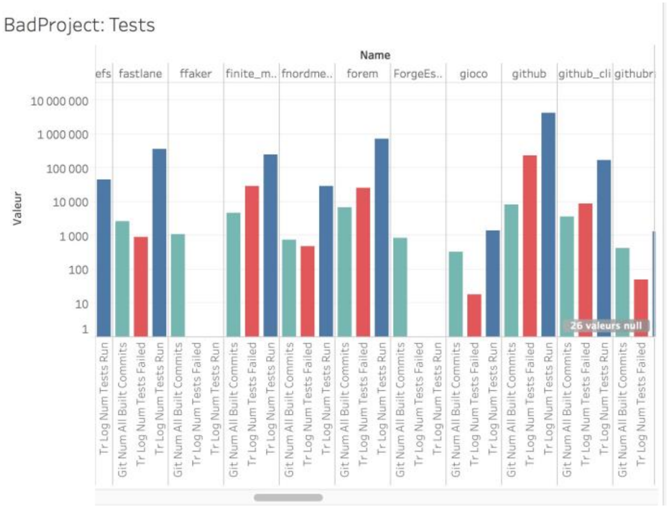

Figure 7 - bad projects

So we could think that the number of tests, successful or not, has some influence on the quality of the project.

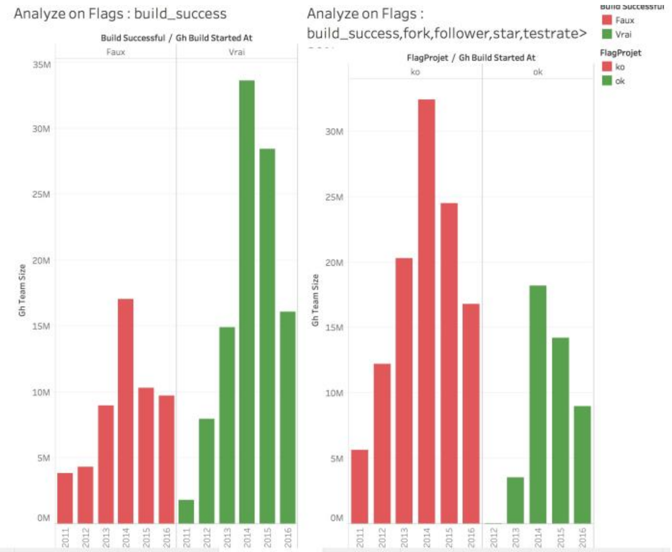

Figure 8 – team size

On our last image, we can see on the left the total of team members for failed builds (red) or successful ones (green). So we might think that the success of a build depends on the size of the team, excepted for 2011.

But on the right, we added a condition : each successful build is true (green) only if 30 % of all the tests were successful as well. And here we see the contrary : the bigger the team, the more failed builds for the project.

V. Conclusion

In this project of Software Analytics, we found some surprising results, and some other more logical. With some more time, we could proceed to a deeper analysis, and get some more conclusive data, but the little we had is already very interesting. Without our last graphic for instance, we would never have thought that a big team size was a problem for a successful and good project.

With Tableau, we did descriptive analysis, but with some more time, we could also have used machine learning algorithms to process our data, and this discipline might help us finding more interesting correlations.

When we started the project, our main questions were :

1°) What makes a good project ?

We found that the criteria for a good project lies in the number of stars, followers and forks. When the project has these characteristics, it seems to be a rather big one.

2°) Is there a relationship between the tests and the quality of a project ?

It seems that yes, good projects have been more tested than the bad ones. But for some projects, we could see a lot of followers, though there were not so many tests on it.

VI. References

TravisTorrent
https://travistorrent.testroots.org

Github tutorial
https://git-scm.com/book/en/v2/Git-Branching-Remote-Branches

Docker
https://docs.docker.com/get-started/#conclusion

Tableau SoftWare
https://www.tableau.com/learn/training#getting-started

Basic javascript
https://github.com/mikedeboer/node-github

VII. Annexes

SoftWare_Analytics_Project.zip
Includes
- docker-compose.yml
- docker-images