# Finding Traces of the Development Process in Travis CI Data

Anonymous Author(s)

**ACM Reference format:**
Anonymous Author(s). 2017. Finding Traces of the Development Process in Travis CI Data. In *Proceedings of 3rd International Workshop on Software Analytics, Paderborn, Germnay, 4 Semptember, 2017 (SWAN2017),* 4 pages.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Next to humans interacting with machines, Software Engineering also relies on humans interacting with other humans. These social interactions are relevant for process analysis and are best evaluated by means of social sciences.

During software development a huge amount of data is collected. This data can be analysed by methods from obersvational research to draw conclusions about the processes that produced the resulting software. The collected data needs to be clustered into test groups before if it can be analysed using these methods.

We analyse the TravisTorrent as an example to find traces of the developent process as follows:

- We describe how to apply methods of social sciences to big datasets (Sect. 2).
- We do a classification of different potential test groups and discuss the value of the remaining data (Sect. 3).
- We show how additional features can be identified by extending the dataset (Sect. 4).

## 2 BACKGROUND

Software Engineering can be seen as a process framework where a set of activities and patterns are carried out by humans to produce a software [1]. This leaves much space on how the actual process flows are desgined resulting in potentially very different approaches. The scope and level of detail of applied processes can differ significantly leading to completely different processes that are hardly comparable.

Recent research puts an increasing focus on the social interactions within software development processes. This results in methods of social sciences being applied to Software Engineering research. Ralph et al. [2] apply *Action Research* to analyse the engineering practise in a single project. This research allows for direct feedback to improve the process observed. The potential to make general observations from the study is limited.

To draw more general conclusions, observational methods need to be applied. A meta analysis of a broad range of similar studies can be done. This is costly and time consuming. Big datasets that record a lot of information about the development of individual software products are broadly available for analysis. This data can be clustered to reflect different features of Software Engineering practice and especially different software development processes. The resulting subsets can then be seen as test groups and analysed with methods also applied in the evaluation of social studies.

We have analysed the TravisTorrent [3]. This dataset collects information about Continuous Integration (CI) results of different project hosted on GitHub. All data is merged into a big dataset to faciliate the analysis. The goal is to provide a starting point to analyse CI processes and their impact on software quality.

The focus of TravisTorrent is the testing phase of Ruby and Java projects hosted on GitHub. Most of the information about the underlying process is left out. The dataset is pseudonymized at the level of commits. It is not even possible to identify single developers without additional data. Even thought each project has some kind of common change process, this assumption is not very precise. On the one hand, GitHub groups could have a very strict process for all their projects. On the other hand, even within a project there might be different processes for each branch.

The testing phase of Continuous Integration can have a different focus depending on the process framework applied. The range includes but is not limited to test driven development, system testing, or testing to avoid regressions. Any or all of these test strategies can be found in a single project. The role of the tests in the project cannot be left out when comparing effectiveness of automated tests within CI. Thus, CI must be seen in the context of the applied change process.

## 3 TEST GROUPS

To be able to evaluate a feature of a recorded process, the data needs to be classified by the different possible characteristics to build test groups. This limits our research to features that are included in the dataset under consideration.

The data has been downloaded and analysed using a Jupyter Notebook[1] [4] with Python and pandas [5]. For each build id in the dataset, only the first row was kept for further analysis. To get a confidence level of 0.9 with a confidence interval of 0.033, a minimum of about 625 samples is needed[2]. This was used as a lower boundary for the minimum number of data rows to allow a test group to be valid.

The dataset itself is already focused on projects written in Java and Ruby [7]. However, when it comes to homogenous test groups, this over-simplified classification is not enough. Clustering the dataset by project significantly reduces the available data. Many projects have only little amount of data available as can be seen in Table 1. Only 273 of the 1283 projects monitored have more than 600 data rows available for detailed analysis.

*SWAN2017, 4 Semptember, 2017, Paderborn, Germnay*
© 2017 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1]The Jupyter notebook used to analyse the data can be found at *anonymized*

[2]This value is calculated using the formula $s_{sample} = \frac{z^{\star}}{2 \times c_i}$ with $z^{\star} = 1.645$ for a confidence level of 0.9 and the confidence interval $c_i = 0.033$ [6].

**Table 1: Number of projects with a given number of data rows.**

| | Number of | | |
|---|---|---|---|
| Data rows available | *projects* | *groups* | *branches* |
| less than 60 | 27 | 11 | 52567 |
| 60 to 120 | 195 | 139 | 430 |
| 120 to 250 | 403 | 294 | 480 |
| 250 to 500 | 318 | 241 | 317 |
| 500 to 1000 | 183 | 151 | 159 |
| 1000 to 1500 | 70 | 69 | 47 |
| 1500 to 2000 | 29 | 28 | 24 |
| more than 2000 | 53 | 68 | 28 |

## 3.1 Classification on Organizational Structures

A first and common classification is to use the organizational structure. Everyone who contributes to a certain project most probably has adopted the respective development and change process prescibed by the project. This can be further generalized to GitHub groups that might share a common development process. In contrast, different processes might be applicable on different project branches.

This does not allow much assumptions about the applied change process. We can only make vague assumptions based on the bare usage of TravisTorrent as a service. In addition, single branches need to be seen in context. If you follow a "stable trunk" pattern it is vital to fix failed builds on the master branch. There might also be discontinued branches or branches that do not have CI at all. This is also highly dependent on the overall change process.

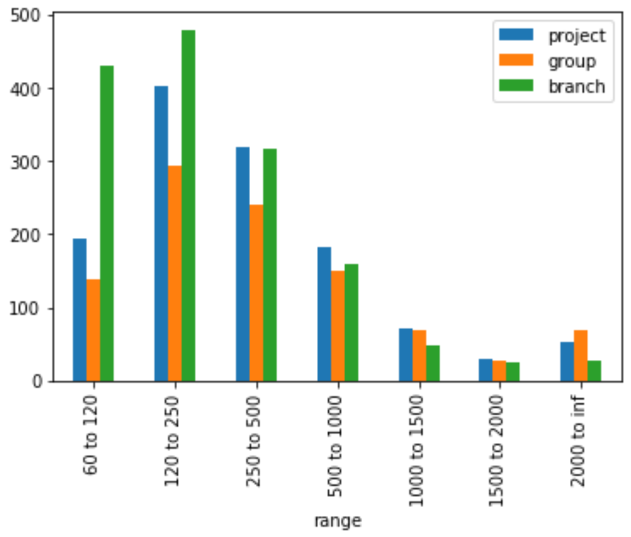

**Figure 1: Sample sizes after classification by project, group or branch.**

Figure 1 shows how the sample sizes change if the dataset is classified by group or branch instead of by project. Especially the classification by branch shows a significant reduction of the sample sizes available.

## 3.2 Classification on Derived Data

After first analysis, we derived additional information based on the data available. Using the build's timestamp the time of day and the weekday when the build started can be identified. The time of day does not appear helpful as all dates are converted to UTC. This inhibits analysis such as whether builds are more stable at a certain time of the day (e.g., within or outside office hours).

The day of week can be used to identify projects that commit only on weekdays. Those projects are most probably driven by inner worktime employees. Figure 2 shows the sample sizes by projects for projects that only commit on weekdays.

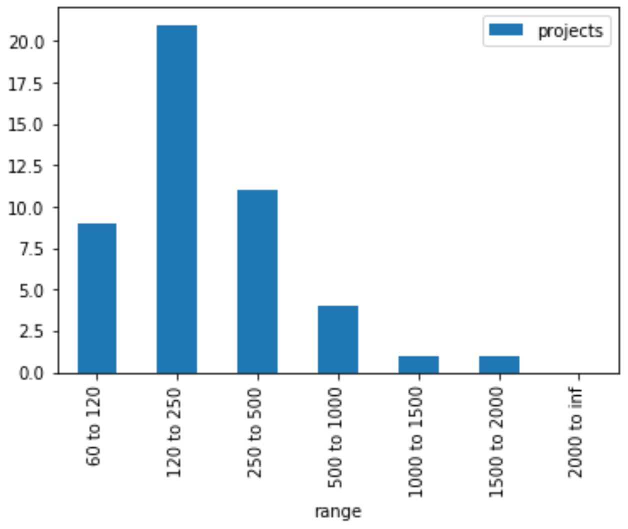

**Figure 2: Sample sizes for projects where builds are only run during the week (Monday to Friday).**

In contrast we cannot make safe assumptions about projects that have builds distributed over the whole week. However it is more probable that they have a broader range of developers including volunteers and overtime workers. Figure 3 shows the available sample sizes for those projects.

Comparing Figure 3 with the general distribution in Figure 1 a very similar distribution can be found. This changes in Figure 2. That might be an effect of the much smaller total number of datasets. It is not surprising to see that most projects have builds through all the week. GitHub focuses on OpenSource projects that often have mixed teams with volunteers and employed developers.

Another attempt was taken to classify the projects by the duration of their broken build phases. A test-oriented development process focusses on fixing failed CI builds. Only a few commits is expected in between the first failure of a build and the fix makes it stable again. A longer phase of successive failed builds thus indicates a process less commited to fixing broken builds.

To accomplish this, all successful builds were identified that have a predecessor that did not succeed. Starting from those builds, all

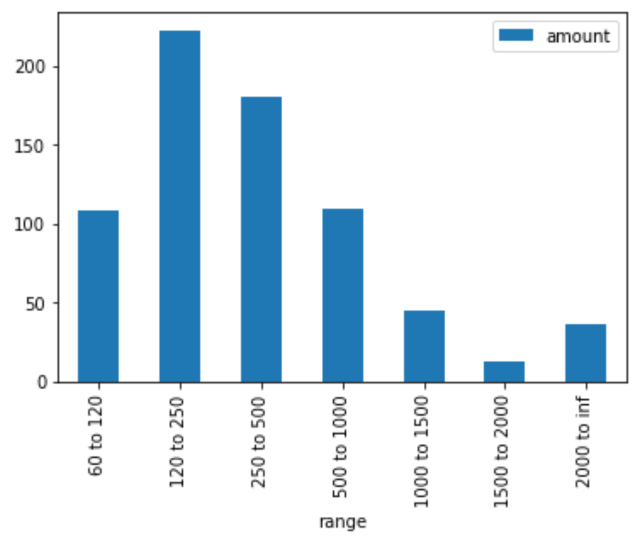

**Figure 3: Sample sizes for projects where builds are only run all week including weekends.**

predecessors are collected. This is continued until another successful build is found or until we ran out of data. The data analysis was not yet completed. The dataset stores the build chains only per project but not per branch. The applied reconstruction technique is not enough for this condition. Every build of a stable branch terminates a broken chain leading to wrong results as displayed in Figure 4.

### 3.3 Control Group

A control group is completely lacking. However, this is something Big Data analysts have to cope with (see [8]). In machine learning you randomly split your dataset into a training and a verification part. This has some properties of using a control group but does not fully get the point as you cannot derive any evaluation on whether a change in process was good or bad.

Another approach to test the validity of data and increase the reliability of the results is Grounded Theory (GT) [9]. Methods like "Dynamic Grouping/Sampling" are especially well suited for big datasets. A random pair of datasets is selected and their assignment to the test groups are switched. Now the study is evaluated again. This process is reapeated several times and allows for identification of relevant features of the test groups.

## 4 FEATURE IDENTIFICATION

The usual approach of studies in social sciences is to define the features of interest and then carefully assembling test groups. This is not possible for a pure data mining approach. Instead, additional afford needs to be done to extract features that are already documented by the collected data.

### 4.1 Combination with other Datasets

One can use GHTorrent[10] to add information about individual developers to the commits documented by TravisTorrent. On the one hand, this leads to observations that reflect the individual traits of each developer. It might also allow to derive more information about the implied development process. On the other hand, conclusions about the team and interactions between team members could be drawn. As a basic example, it would allow to identify the number of developers contributing to a single project.

### 4.2 Feature Mining

By further mining the dataset, additional features could be identfied. A combination of available features could also lead to new indicators. Extending the dataset could clarify some of the uncertainties limiting the test group identification from Sect. 3.

Some data can also be subject to further mining activity. For example natural text recognition can be applied to extract information from commit messages. Similarly, information can be extracted from the commited artefacts.

Of course, more data can be collected for future analysis. As features of interest are identified data can be collected that allow characterization of the data. As software development is a highly dynamic process a lot of data points can be raised in a short time.

### 4.3 Provenance

While lots of data is easily available, especially when it gets to process information data acquisition is incomplete. By recording the provenance [11] of the software development process, this could be augmented. Models are available to describe the provenance for Software Engineering processes [12].

A lot of information are already available within existing datasets. Some information might easily be acquired. But information about processes that cannot be automated is hard to record.

## 5 RESULTS

To apply the methods of social studies, only one characteristic may be changed within a test group compared to the control group. When analyzing the TravisTorrent, we can identify different features to form our test groups. The more precise the actual characteristic of the feature can be identified, the better we can evaluate processes.

Some of the characteristics that we identified can already give a pretty good feature identification like filtering by branch or project. Also more complex characteristics like weekend activity are possible to identify. However, the confidence in the dataset is reduced with the assumptions that are made.

The dataset itself shows some deficiencies in the choice of projects, reduction of information, and clearity. However, additional datasets can be used to compensate for that. Overall the dataset sizes reduce a lot when looking at certain features. Yet there is still enough substance for further analysis.

Another way is to enrich the dataset with additional information to allow better classification or collecting additional data. Adding more details about the underlying process framework is complex. Processes are not documented in a standardized way and information retrival will most probably be a manual task. Project hosting platforms like GitHub might be helpful by providing information about the process automation applied by the projects.

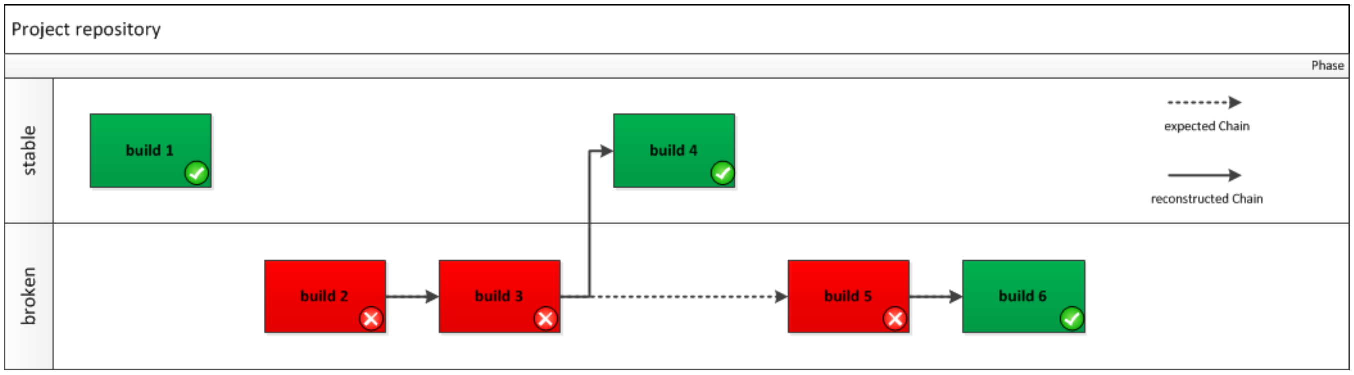

**Figure 4: Example of an incomplete reconstruction of a build chain.**

Acquisation of additional data with the focus on social science is the most promising approach. As research on this field gets incresing attention, new datasets can be expected. Identifying features of interest and evaluationg the collected data should be coordinated with newest research from social sciences.

## 6 RELATED WORK

As the TravisTorrent provides the base for the MSR 2017 Challenge, much work was produced on this dataset. The GHTorrent was the base of the TravisTorrent. It is therefore a great source to extract more information about the process that leads to testing [3, 10].

Some information can be found about Software Engineering within the context of social sciences. Ralph et al. [2] outline that common practice on software engineering research should be thought of as action research. The researcher does not only integrate into the team but tries to identify reactions to certain modifications in the process.

Big data in contrast provides observational data. This again enables explorative research unsing quantitive means. Sampling techniques from this field provide possibilities to increase the reliability and validity of the data in use [13]. Further mining techniques like natrual language recognition can be applied to the data.

It is not an easy task to capture processes. Code hosting platforms such as GitHub provide the backbone to support and model processes. To evaluate if humans adhere to those processes, it's provenance [11, 12] provides a good starting point.

## 7 CONCLUSIONS

Using this approach, one can in fact do more observational research as the process is completed by the time the data is analyzed. The set of usable obervations can be extended only in a limited way. You still have no control group at hand to evaluate the result of certain changes.

We tried to cluster the TravisTorrent dataset into test groups as a first step towards analysis of the data with methods from social scienes. We soon realized that the data was not sufficiant to make fair assumptions about the test groups.

The dataset could be augmented and additional data can be collected in a way that well defined test groups can be identified. The collection of provenance data about the development process would further enhance the possibilities of classification of the data presented in the TravisTorrent or similar datasets.

## REFERENCES

[1] R. S. Pressman and B. Maxim, *Software Engineering: A Practitioner's Approach.* McGraw-Hill Education, 8 ed., 1 2014.
[2] P. Ralph, M. Chiasson, and H. Kelley, "Social theory for software engineering research," in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, EASE '16, (New York, NY, USA), pp. 44:1–44:11, ACM, 2016.
[3] M. Beller, G. Gousios, and A. Zaidman, "Travistorrent: Synthesizing travis ci and github for full-stack research on continuous integration," in *Proceedings of the 14th working conference on mining software repositories*, 2017.
[4] T. Kluyver, B. Ragan-Kelley, F. Perez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, and J. D. Team, "Jupyter notebooks – a publishing format for reproducible computational workflows," in *ELPUB2016. Positioning and Power in Academic Publishing: Players, Agents and Agendas, 20th International Conference on Electronic Publishing, 7-9 June 2016 in Göttingen, Germany*, 2016.
[5] W. McKinney, "pandas: a foundational python library for data analysis and statistics," in *PyHPC 2011: Workshop on Python for High Performance and Scientific Computing, SC11, Seattle, WA, USA, November 18, 2011*, 2011.
[6] J. Bortz, *Statistik für Human- und Sozialwissenschaftler.* Berlin: Springer, 2010.
[7] M. Beller, G. Gousios, and A. Zaidman, "Oops, my tests broke the build: An analysis of travis ci builds with github." Submitted to PeerJ, 2016.
[8] J. J. Berman, *Principles of Big Data: Preparing, Sharing, and Analyzing Complex Information.* Morgan Kaufmann, 2013.
[9] B. Glaser and J. Holton, "Remodeling grounded theory," *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research*, vol. 5, no. 2, 2004.
[10] G. Gousios, "The ghtorrent dataset and tool suite," in *Proceedings of the 10th Working Conference on Mining Software Repositories*, MSR '13, (Piscataway, NJ, USA), pp. 233–236, IEEE Press, 2013.
[11] L. Moreau, P. Groth, S. Miles, J. Vazquez-Salceda, J. Ibbotson, S. Jiang, S. Munroe, O. Rana, A. Schreiber, V. Tan, and L. Varga, "The provenance of electronic data," *Communications of the Acm*, vol. 51, no. 4, pp. 52–58, 2008.
[12] H. Wendel, M. Kunde, and A. Schreiber, "Provenance of software development processes," in *Provenance and Annotation of Data and Processes: Third International Provenance and Annotation Workshop, IPAW 2010, Troy, NY, USA, June 15-16, 2010. Revised Selected Papers* (D. L. McGuinness, J. R. Michaelis, and L. Moreau, eds.), (Berlin, Heidelberg), pp. 59–63, Springer Berlin Heidelberg, 2010.
[13] M. Patton, *Qualitative research & evaluation methods : integrating theory and practice.* Thousand Oaks, California: SAGE Publications, Inc, 2015.