Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

CSU33012 Software Engineering
Software Engineering Report

Daniel Ilyin, Std# 18320428

Contents

1 How to measure engineering activity

1.1 Source lines of code

The most 'obvious' and one of the earliest methods of measuring an engineer's activity is Source lines of code (SLOC). SLOC comes in 2 main types, Physical SLOC and Logical SLOC.
Physical SLOC is a count of lines of text in a programmes source code, which can sometimes include the comments too. Logical SLOC on the other hand tries to count executable expressions, but their specific definitions depend on the coding language.
Although it has its uses, SLOC it is a terrible way to measure an individual engineers productivity. It has been attempted in the past, and engineers would benefit from writing stretched out functions with unnecessary code, to increase the perception of productivity.
One flaws of using SLOC to measure an individual engineers productivity is that some coding languages tend to be more verbose while others are more efficient which could lead to false conclusions such as engineers using a static language, such as Java have double the SLOC of engineers using a dynamic one like python.
Another flaw is that implementations of certain logic can differ depending on the experience of the engineer, and an experienced engineer could implement a feature in significantly less lines of code than an less experienced one, and by using SLOC it would seem that the less experienced engineer has more engineering activity and a higher productivity.

1.2 Commits

Another way that we can measure an engineer's activity is by counting the amount of commits they make towards a repository, and looking at how often a commit is made. Most version control providers offer an easy way to see how many commits an engineer has made within a time period, and how often they commit code which can then be compared with the other engineers that worked on the repository. This, on it's own isn't a very useful metric as a commit can be as little as 1 character change due to a typo, or can be as large a adding a completely new feature.

2 Platforms used

Due to the vast amounts of data that need to be considered when trying to measure an engineers productivity, there are many start-ups that have become very successful in creating engineering management tools.

2.1 Github

Github is the most widely used version control and source code management system using Git. Multiple users can work on the same project at the same time which is held in a repository, without worrying about writing over or interfering with another users commit. Engineers can branching repositeries while implementing new features without disrupting the Master branch and can merge back onto the master branch when everything is complete and working. Github also offers a lot of metrics that can be analysed on the site itself per user and also per project, with more in-depth metrics being available through the Github api.

2.2 Waydev

Waydev is an engineering management tool, which logs commits and pull requests from Git data, as well as offering informative insights and reports. It offers daily updates which track the evolution of the team's velocity sprint over sprint. Waydev also offers a team performance report which gives every engineer an impact score, efficiency percentage, SLOC, commits submitted and other metrics. Having this data available to view to each engineer and not only management help's motivate each engineer to stay on track and to keep a good performance and efficiency score.

2.3 Pluralsight Flow

Pluralsight Flow is an engineering platform, similar to Waydev, which collects Git data into easy to understand insights and reports. These insights are visualised in the dashboard and give important feedback based on the data it collected, such as how much time is spent refactoring legacy code vs new code, suggests possible bottlenecks within a project and provides concrete data on commit risks. Flow also provides useful visualization of the engineering team's code review, which helps with spotting unreviewed Pull Requests. Engineer's are encouraged to contribute meaningful commits, as Flow can estimate the impact of every commit, which can be viewed by each engineer, also increasing accountability and competition within the team.

3 Algorithms

3.1 Simple counting

Simple counting involves counting a certain metric such as SLOC or commits, and assigning an engineer a score based on the metric. This method can be useful as a quick estimate on an engineers activity but clearly has its limitations as I've touched on before. An engineer having a lower amount of SLOC than other engineers doesn't have to mean that they are less productive but can imply that they are more efficient in their code. An engineer commiting code twice as much as their peers can imply that they are cleaning code, or fixing small typos or bugs, or adding comments. This is why we need to measure multiple metrics with each metric having a different weight, to monitor an engineers productivity more accurately.

3.2 Machine Learning

Machine learning has the ability to study large datasets of engineers' metrics and over time, develop a good understanding as to how each metric should be weighted, based on how each metric affects engineer's activity and productivity. Over time this algorithm can then be used to give out better and more informed productivity and efficiency scores to engineers.

4 Ethical dilemma

Privacy is a key area of concern. An employer tracking an engineers productivity, involves a lot of sensitive and personal information, and although GDPR is very strict with the handling and processing of this information, if the data collected was to be leaked, the consequences could be devastating. This would not only hurt the relationship between employee and employer, but could lead to bigger problems to the Company itself.

Another area of concern is an employee can feel huge stress knowing that they are being heavily monitored by management, and if they prefer to work untraditionally while their team works traditionally, their metrics might show them as underperforming even if they are putting in the same amount of work, and outputting what they need to be.

I think that employers should be analysing the productivity and performance of their employees but there has to be done ethically and there should be a fine balance between monitoring and micromanaging. The employees should not feel that they are being watched every second of the workday, but should also be keeping up with their duties.