**Measuring Software Engineering Report**

**CSU33012**

**Conor Doherty 17322281**

Introduction

The technology sector has been one of the world’s fastest growing industries in the past twenty years. Software engineers are some of the most sought-after employees, however this role is unique in how difficult it can be to measure developer performance. In other sectors such as finance or law, employers could simply monitor efficiency by individual profit generated or cases closed but this is a more challenging task when it comes to assessing a software engineer’s output.

In a world where the rise in agile development necessitates a quick transition from development to product testing to shipping the product, project managers must be able to understand the capability of their team to meet deadlines and handle the required workload. By collecting data on developers, company leads can measure the productivity of both teams and individuals by passing this data through algorithms which help to grade performance during a project and forecast future potential time required for producing a product. This can lead to team members being moved around or assigned to different areas, ensuring even distribution of workload and proper checks for quality assurance.

Proper data collection and analysis can be crucial for the growth of both companies and their employees. Ensuring that teams are operating at peak efficiency means companies can maximise their profits and produce products in a quick and thorough manner. Time taken to review coding practices and the project process pays dividends in the long run. The same can also be said for individual reviews in accordance with gathered metrics. Employees can look back on data with their superiors and use that to identify skills they may need to acquire through extracurricular learning, ways they can better use their time during product development and how they can function better within a team. This would allow the teams the employee is a part of to function better and allow the employee to progress as a developer, which is also ultimately beneficial for the company.

Engineering Activity Measurement

Useful and attainable metrics must be chosen before any algorithms or frameworks are written to measure productivity. In the early days of software development, developers were assessed by how many lines of code they had written. This led to inaccuracies such as the unfair comparison between those writing in machine code and in higher level languages and did not account for bugs or complexity in the code. The first attempts towards goal driven metric activities were made by Basili and Rombach to compare the coders output against specific objectives. (Fenton and Martin: Journal of Systems and Software 47). Today one could observe the number of commits to a project repository an engineer has made or how many issues they resolved on a project board however this does not account for the difficulty of the project, the quality of the commit or any collaborative efforts made between multiple team members. In modern Agile development, Unit Testing and quality assurance can take up a significant amount of time near the completion of the project so this equally important contribution must be considered.

Business management must also decide how to collect the data required for analysis. Some frameworks may have built in functionality for this but if you are starting from scratch then you must decide whether to collect data in real time or to log the developer’s actions and refer to them at a later date. For both options, the employee’s privacy and happiness must be considered. The most important thing when gathering any metric is that they are easily comparable, attainable and accurate. If possible the data should also be free from bias, so care must be taken if data is based on opinion or feelings.

Platforms for Measuring Software Engineering

There are many platforms and frameworks available for measuring the productivity of software engineers. The goal of these platforms is for a business increase output of their developers in terms of commits and codebase impact while maintaining a high quality of product. One such platform is Pluralsight Flow. Their platform allows a team leader to identify how much time is spent writing new code compared with updating legacy code, negate bottlenecks and minimise risk for significant commits. There are also code review visualisation resources to ensure reviews are helpful and interactions between the team are healthy. Their website claims the platform enables a 21% increase in coding days per week, a 25% increase in commits per day and a 20% increase in codebase impact. The company boasts customers such as AWS, Github and Google. While these percentiles may not directly represent increased output by developers, they suggest that the previously mentioned tools to gather and visualise data should improve team cohesion and the capacity of individuals to maximise their time during a project.

Another possible platform to avail of is Waydev. This platform gathers data through the git infrastructure already present in a company. Waydev allows teams to communicate their progress and potential pitfalls to business leaders, identify how project plans are affected by bug fixing and scope creep and generate flexible reports. Engineer output data is automatically tracked in line with agile methodology. Waydev also has daily stand-up functionality as well as individual meetings and code reviews, all of which gather and utilise metrics to improve decision making and improve team yield. Their website states 32% faster code deployment, 24% increase in feature delivery and 21% less unplanned work, which has been enough to entice fortune 500 companies such as Caterpillar and Jolt.

A third platform a business could use is Velocity from Code Climate. Velocity uses analytics to negate bottlenecks and increase visibility of development progress which allows teams to decrease time needed to deliver a product. This involves tracking merge risk and version controls visualising data sets by team or individual, setting tasks and monitoring scope creep, analysing code reviews, and assessing team member skills to ensure they are placed appropriately in the project.

Each platform mentioned above, as well as any other frameworks available, have a range of strengths and weaknesses. The impressive clientele and strong measurable improvements brought to these businesses show that these services help team to increase their efficiency. By gathering this data and visualising it, team leaders can make meaningful steps towards enhancing product creation and use their team members to the best of their ability. One downside to using these platforms is that they will not necessarily compatible with the git setup currently present in the company. This could be of detriment to output for the period of time it takes to streamline the platform into the business, or else limit the choices team leaders have in choosing a platform.

Types of Computation Over Data

There are many calculations one can do to with software engineering data, each coming with advantages and disadvantages. As previously mentioned, a very basic metric for measuring developer output is to count how many lines of code they produced. Productivity can be defined as output quantity/period of time (Emmanual Weiss). Therefore, a baseline calculation we arrive at using simple data is Lines of Code (LOC) per person month. An advantage of this calculation is the data is easily collected and comparable between employees. However, there are myriad disadvantages, including incentivising inefficient code and discounting recycled code, comments and the difference in complexity of languages.

A more widely used and current calculation for measuring efficiency is process cycle efficiency. This algorithm is used across a range of industries but applies itself perfectly to calculating developer efficiency. It is measured by value added time/ cycle time. Value added time is time spent adding value to the product. This includes all time spent working on the product, including writing code, testing and quality assurance. It does not include necessary activities like an employee’s lunch break or time spent on other projects or any other wasted time. The cycle time is the total time spent developing the product. To use this calculation, a standard must be put in place throughout the teams gathering the necessary metrics to record their time spent accurately and maintain that standard across projects. Process cycle efficiency is a more useful representation of employee efficiency, but it relies on employees being truthful with their reporting of time usage.

Another useful software engineering specific algorithm is the percentage of time spent on planned work. This would simply be calculated by time spent on planned work/time spent on planned work + time spent on unplanned work \* 100. Unexpected bugs and problems pop up frequently during the development of projects which are necessary to fix. This can lead to a struggle to meet deadlines and halt progress in creating other areas of the product. By calculating how much time a developer is spending on planned work, a team leader can see whether a member is struggling with a lot of obstacles, a lack of focus or time management if their score is low or allocate more responsibility if their score is high.

The Ethics and Legality of Data Collection for Optimisation

The ethics of tracking employees at all times are dubious in my opinion. Companies must put some trust in their workers to do a good job and if data collection is necessary then there must be consent from the employee, an option to opt out and no discrimination against those who do not wish to take part. Under GDPR, employees have a right to information about the collection and processing of their personal data, access to the data held about them, have their data erased, restrict this data being processed and data portability. These rights must be taken into account if an employer wishes to collect any data in the workplace.

Employers must also be wary of the safety of data being collected by external platforms such as those mentioned above. In July 2020, Catalin Cimpanu reported on ZDNet.com that Waydev had experienced a security breach. Waydev customer’s Github and GitLab OAuth tokens were stolen which allowed the hackers access to their Github accounts. The breach was detected by Github through suspicious activity on a Waydev users Github account and raised the alarm. This highlights the responsibility that employers have not just with the data that they collect but also with the platforms they entrust with their employees’ data, as the blame for any breach does not only fall on the company with the vulnerability but also with the company who decides to work with them.

Employers must also consider the stress that constant monitoring would place on employees. Google utilises a mix of qualitative and quantitative data (Laszlo Bock) acquired through employee metrics and person to person reviews. In my opinion to counteract a higher than average number of reviews and data analysis, they implemented the laid back company culture of bean bag chairs, free cantines and nap pods that is emulated by so many companies today. Constant data collection could create a “Big Brother” atmosphere, so companies try to distract employees with a “fun” workplace. Team leader must also ensure that the focus on efficiency does not create such a stress on the developer that it lessens their work, which would defeat the whole purpose of measuring efficiency. This atmosphere can be seen in Amazon fulfilment warehouses. The Amazon office workers are treated well, with similar perks to Google employees, however warehouse workers are subjected to constant surveillance, where the number of items they load per hour is tracked, toilet breaks are disincentivized and Artificial Intelligence is used to ensure social distancing protocols. In a Washington Post article by Jay Greene, it was found that Amazons serious injury rates are nearly double those seen in warehouses run by other companies. This demonstrates that any data gathered by Amazon is not to enforce safety standards, it is simply to push employees to their limits and increase profits for the company. This same ethos must not become the norm in any industry, including within the scope of measuring developer efficiency.

Conclusion

From my research for this report, I can conclude that there is no simple metric or calculation which can be used to represent Software Engineer Activity. This is an industry which has been constantly evolving for the past fifty years with new advancements and jobs being created within it, thus making older methods of quantifying their respective output obsolete.

I think that the onus is on industry leaders to protect their employees and their data in the quest to maximise their profits. In the post-privacy era, team leaders must tread carefully in obtaining useful, consented, accurate data which then must be stored safely and only shared with those permitted to see it. These metrics should be used to calculate ways to improve the customers experience by delivering a better product faster without being of detriment to the wellbeing of developers.

References

Fenton, N. E., and Martin, N. (1999) "Software metrics: successes, failures and new directions." Journal of Systems and Software 47.2 pp. 149-157.

<https://www.pluralsight.com/product/flow>

https://codeclimate.com/

<https://www.zdnet.com/article/hackers-stole-github-and-gitlab-oauth-tokens-from-git-analytics-firm-waydev/>

Measuring LOC and other basic measurement Seminar on Software Cost Estimation, WS 2002/03 (Emanuel Weiss)

APPROXIMATING THE PROCESS CYCLE EFFICIENCY OF NON-PHYSICAL PRODUCTION SYSTEMS (Chang-Sun Chin)

Measuring Performance in Large Scale Agile Software Development Projects A Case Study at Saab (Evelina Magnusson and Moa Westlund)

<https://www.citizensinformation.ie/en/employment/employment_rights_and_conditions/data_protection_at_work/data_protection_in_the_workplace.html>

Work Rules – Laszlo Bock

Amazon’s employee surveillance fuels unionization efforts: ‘It’s not prison, it’s work’ – Jay Greene, Washington Post