Measuring Software Engineering – Report

Davy O'Leary-Fraad 19334296

Contents

Introduction

Software engineering is defined as the process of designing, developing, maintaining, testing and evaluating computer software with respect to the principles of software engineering itself. The past decade has seen numerous advances and innovations in technology, with the driving force behind this being the development of new, powerful software applications. It is not surprising then, that software companies have been eager to recruit new developers, and ensure that current employees are operating as efficiently and productively as possible. This is where the topic for this report originates. These companies require a vast toolset in order to gather and analyze these massive data sets. The objective of this report is firstly to explore the ways in which one can measure the activity of a given engineer. Secondly, I will discuss the platforms and applications that companies and individuals alike are using to perform this data mining and compilation. Thirdly, I will cover the various algorithms and processes that are used to profile the performance metrics into a more accessible and visual form. Finally, I will be discussing the ethical and moral issues surrounding this type of data collection / surveillance. Hopefully, through writing this report, I will possibly be able to gain some insights into this field, and how I, an amateur software engineer, can present myself to companies and potential employers in the best way possible.

Section 1 : Methods of Measuring Activity

Nowadays, it is not a difficult task to measure an software engineer's activity in the workforce. Many technologies are available that enable employers to gather this data with ease. The question is, what data should one gather in order to create an unbiased and accurate insight into an engineer's activity.

There are definitely a variety of aspects to look at, with some being obvious areas for exploration and others possibly more obscure. At a basic level, looking at tangible values such as the number of code commits performed, or the amount of hours spent working, can provide a decent insight into the productivity of an engineer. One can consider comparing the effectiveness of an engineer in individual projects versus while working on a team. This may shed insight into the engineer's optimal work environment. Only observing these surface level numbers can however overlook more important details such as the efficiency or correctness of the code. These variables can too be measured, but they require a bit more work to analyze and are not necessarily figures that can be provided by an application function. A metric that can be used for this kind of analysis is the time and space complexity of the code.

Beyond this, one can also look at more hidden variables. The interactions between an engineer and his team or colleagues can shed insight into his/her productivity.

Similarly, one must consider an engineer's experience and consider that they may be inexperienced in the language or topic of the project they are working on. This will definitely have a large effect on productivity and efficiency. Realizing this data in a visual sense can reveal an employee's strongest areas and weak points alike.

To a degree one should even consider the attitude of an engineer. If an employee has a good attitude towards work and learning, this should be taken in to account, as it will ultimately have an effect on their productivity and willingness to work. However, aspects such as attitude and social interaction undoubtably are much more difficult to quantify, and attempting to do this will likely require invasive and ethically questionable technology. Despite this, I believe that these are definitely things to investigate, on top of simple metrics like code commits. In fact, these are not entirely speculative ideas, as companies are quickly adopting devices and software that can monitor their employees behavior. It may be a more complicated task to visualize this data however, and its possible that employers could misinterpret this information, so it is definitely a task that should require some care and thought. Through the compilation of all this data combined, an employer can create a very thorough and insightful view into the usefulness and worth of their engineers. In the next section, I will discuss the technologies companies are employing to do this.

Section 2 : Technologies for Measuring Activity

Due to the ever growing increase in demand for software engineers and new technologies, the demand for applications and devices that can measure the efficiency and productivity of the engineer has too increased. Not only do devices that explicitly measure this exist, but many tools utilized commonly by engineers in today's workforce collect this data implicitly also.

For example, GitHub is one of the most popular platforms used by engineers today. It is a software that accommodates software development and version control. This means that individual engineers can showcase their work, and teams of engineers can collaborate in a simple and effective manner. Behind this though, is a database that maintains the information for every user on the platform. GitHub employs a REST API that allows users to query the system and obtain data such as number of repositories, code commits and their frequency etc. After retrieving this data, one can easily visualize it in graphical form for simple analysis.

SonarQube is an open-source platform that inspects code quality, detects bugs, even analyses code security. The data is automatically compiled and then visualized, and the platform can even provide useful metrics like evolution graphs. This software can even calculate code complexity and therefore, it provides a very useful environment for the efficiency of a program to be inspected. One useful benefit of

this technology is that it can be integrated into most IDEs. This allows one to gather data while coding remotely, and so can paint a more accurate all around picture of the engineers activity.

Haystack is a widely used product which uses GitHub data to analyze productivity. It is a tool that can be used to identify bottlenecks and trends in mainly team projects utilizing Git version control. This type of software is fairly common, however Haystack appears to be one of the better applications on the market, alongside LinearB. Unlike SonarQube however, this platform works exclusively with GitHub data and therefore cannot be used remotely.

These are some of the technologies that can be used for monitoring formal code and agile process metrics. When it comes to analyzing the developer behavior and team interaction, there is an entirely different toolset available.

Many companies have emerged in the past couple of years, with products intended on documenting employee emotion, behavior and attitude. These products were designed for use in all workforces but obviously can be a powerful tool in a software development setting as well. Healium and Reflect Innovation are two companies that provide this service. For example, Healium, utilizes augmented and virtual reality in conjunction with wearable technology to enable users to ‘see their feelings’. While employees can view this data, it’s clear to see the applications for use in team management also. If employers can view this kind of data, they can see who in their

staff work well together, and who does not. Using this data, employers can pair engineers in order to create the most productive and happy teams.

Section 3 : Computation over the Data

The previous section discussed the platforms available for collecting desirable data. Now I will be exploring the ways in which we can perform computations over this data to accurately profile the performance of the engineer.

One approach that is widely used is the implementation of machine learning and artificial intelligence. With many of the datasets that will be collected, the scale may be so large that it would be impractical and likely impossible for a human to analyze. Therefore, we can implement intelligent algorithms to sift through this data and identify patterns and trends that may have been unrecognizable to the human eye. With machine learning, we can either implement supervised or unsupervised learning. With the former, the designer indicates to the program, the correlation that it should be working to identify, namely, presenting the program with a labelled dataset where the engineer in question was productive and one where the engineer was not. This will allow the program to identify the trends that it is searching for. With the latter, unsupervised learning, a more advanced learning algorithm is required, however, the program will not require a labelled input such as before, and can identify correlating factors on its own.

Simple counting methods may also be implemented on the basis that the dataset in question is not too large and intricate. One may use a method called Object Counting depending on the data being analyzed. This entails identifying objects or parameters that can be easily numerically counted. For example, in a software implementation, this process could literally be used to count instances of an object in a piece of object oriented code, like a Java program. These processes utilize simpler algorithms than machine learning, however can be just as effective given the right circumstances and environment.

Section 4 : Ethicality of these Approaches

As with all forms of data collection, there is always concern that ethical and moral issues may arise. Furthermore, certain pieces of information are more sensitive than others, and one can see how ethics could be called into question depending on the invasiveness of the approach.

One major issue to be addressed is whether the subject of inspection was aware that they were being surveilled and whether they consented to it. Without consent, it is not only unethical but usually highly illegal for a company to collect and store personal data.

On the other hand, plenty of the data that may be used in analysis of engineer activity, is publicly available, through GitHub or a similar platform. This data can

be collected by anyone who wishes to do so, and users of the platform are usually aware and consenting of this lack of privacy. Collection of this type of data should not raise too many ethical issues in terms of consent, however it does raise a new issue.

While employers often use this data to monitor their own employees, it could just as easily be used to inspect potential employees and applicants. While most applicants would be aware of this, it is possible some would not be and therefore, may not have optimized their activity on the platform to reflect the image they would like. As a result of this, a potential employer may make assumptions on the applicant that are not representative of their character or ability. These assumptions may lead to a biased opinion and ultimately a lost job opportunity. While a case such as this is rare, it is definitely possible, and could raise some concerns on the ethicality of the data collection.

When it comes to personal data, such as the collection of information based on a subject's behavior and emotions, a level of caution is required. Employers should seek absolute approval and consent and employees should have the right to view this data, and request an employer deletes their records of it, if the employee so wishes.

Conclusion :

It is clear to see that the landscape of software development and the environment an engineer in today's workplace faces, are changing at a rapid pace. Engineers should be aware of the consequences of their activity, as so much data is easily available, and a lot can result from it. Employers will likely continue to grow more detached from their workforce, while becoming simultaneously more in tune with it. I am interested to see how this situation evolves in coming years, and how it will affect me if I decide to journey into the world of professional software engineering.

References

https://www.sealights.io/software-development-metrics/top-5-software-metrics-to-manage-development-projects-effectively/

https://www.techopedia.com/the-top-6-ways-ai-is-improving-business-productivity-in-2021/2/34505

https://www.hitachi.com/rev/pdf/2015/r2015_08_116.pdf

https://www.youtube.com/watch?v=Dp5_1QPLps0