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

Measuring Software Engineering

Alannah Walsh

17326917

Contents

1. Introduction

The reason for measuring software engineering is that it can improve the process and quality of project development. It can also improve productivity and engagement. This leads to reduced costs and more profits for a company. The enhancement of organisational processes is the future of economic growth in the 21st century. However, measuring software engineering is a non-trivial task and a significant amount of resources are needed. This report looks at the ways in which the process of software engineering can be measured and then analysed. I will discuss the data that can be measured, the analytical platforms to conduct this work, the algorithmic approaches available and ethics surrounding this type of analytics. I will explore each of these topics in turn to give an overview of this extensive subject.

2. Measurable Data

Software metrics are a standard of measure that contain many activities which involve some degree of measurement. They are measures that enable software engineers and managers to gain insight into the efficiency of the software process and the projects conducted using the process framework. The value of applying measurement to different areas is that the engineer/manager can identify where change is needed.

Almost everything can be measured in software engineering, some metrics examples are as follows:

- Lines of code written, or specifically, source lines of code
- Number of functions
- Number of commits
- Frequency of commits
- How fast the task is completed
- Number of keystrokes compared to lines of code written
- Lines of tests per lines of code
- Test coverage
- Code reliability

A software engineer or manager needs to decide which metrics are relevant and effective for their project deliverable and project team. Also, the metrics need to be easy to obtain and useful.

Characteristics of good software metrics:

- Simple and accurate
- Can be used to predict
- Easily obtained and low cost
- Robust, insensitive to insignificant changes
- Valid and relevant to intended measurement

Metrics can be used to increase software development quality and productivity. Software productivity can be defined as the ratio between the functional values of software produced to the efforts and expense required for development. [1]

Productivity can be measured in several ways; two common ways are:

- Size-related which is expressed in terms of the number of lines of code, for example
- Function-related which is expressed in terms of the amount of functional code delivered

Lines of source code per programmer-month (LOC/pm or SLOC/pm) is widely used within the software industry as a productivity metric. [2] This is computed by dividing the total number of lines of source code that are delivered by the total time in programmer-months required to complete the project. The time taken for requirements capture, design, coding, testing and documentation would need to be included in the calculations and the number of lines of source code depends on the programming language used.

However, bear in mind that the programmer who produces the most lines of code is not necessarily the most productive. It may be that the 'less productive' programmer produces better functional and more reliable code. If individual engineers are measured solely on SLOC/pm then they may compromise on quality in order to become more 'productive'. Therefore, there is a need to consider other information about the quality of the programs that are produced.

The benefits to using metrics are threefold:

1. Improved Performance

Productivity metrics measure developer productivity objectively and can be used to communicate expectations to the development team without bias. The metrics help identify and eliminate the factors that hinder the effectiveness of the team which leads to a higher-performing team.

2. Better Outcomes

Measuring and analysing the data collected helps identify issues that can be addressed to improve product life cycles. This will provide the opportunity to deploy better code, faster and cheaper.

3. Project Planning

A project manager can gain better insight into a project and predict the cost and schedule with higher accuracy. Productivity estimates can also be used to inform investment decisions or to assess whether process or technology improvements are effective.

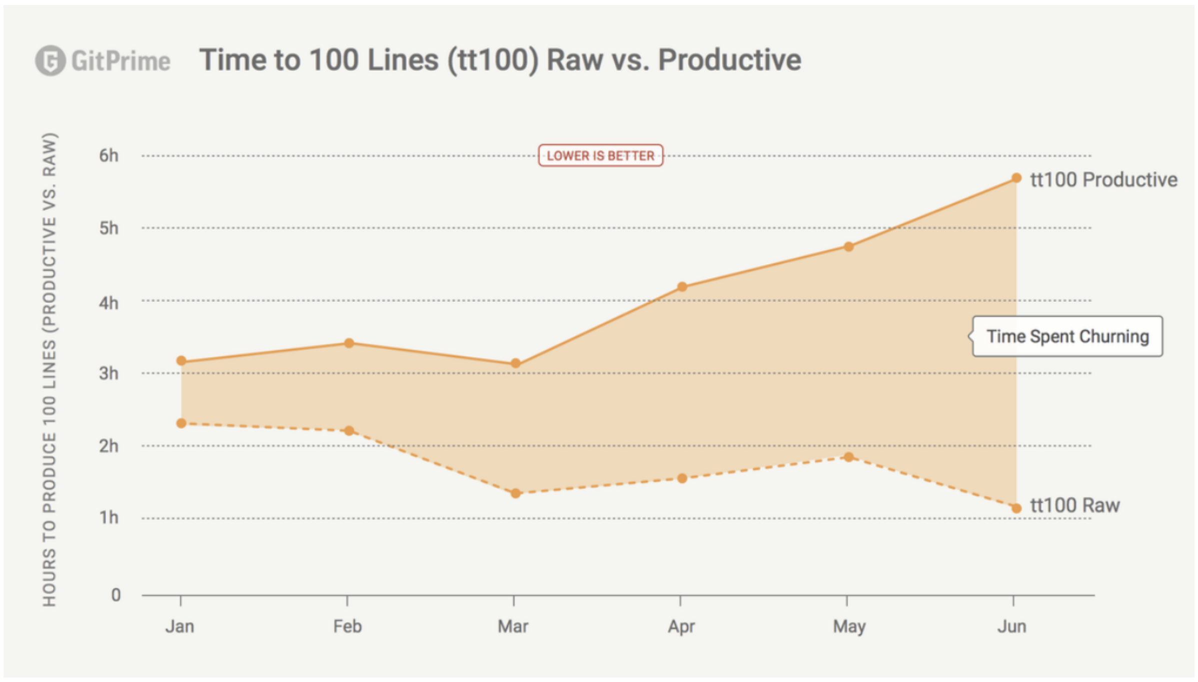
3. Computational Platforms

There are many computational platforms and systems that aid in gathering metrics and the analysis of software engineers and engineering processes. I will discuss GitPrime, Velocity, Hackystat and Personal Software Process.

3.1 GitPrime

GitPrime is a platform which focuses on the analysis of source control, specifically Git. It analyses data gathered from the version control system being used by a company or individual. It analyses a large number of variables surrounding the code being produced and the type of interactions between engineers in pull requests. By doing this they claim to be able to provide insights into the performance of a specific engineer or a team over time. The main incentive of GitPrime is how easily understandable the data is. It visualises things such as pull requests, commits, team dynamic graphs, team's contributions and work habits. You can clearly see all the metrics and understand how the team is performing.

An example of data gathered on GitPrime is shown below:[3]

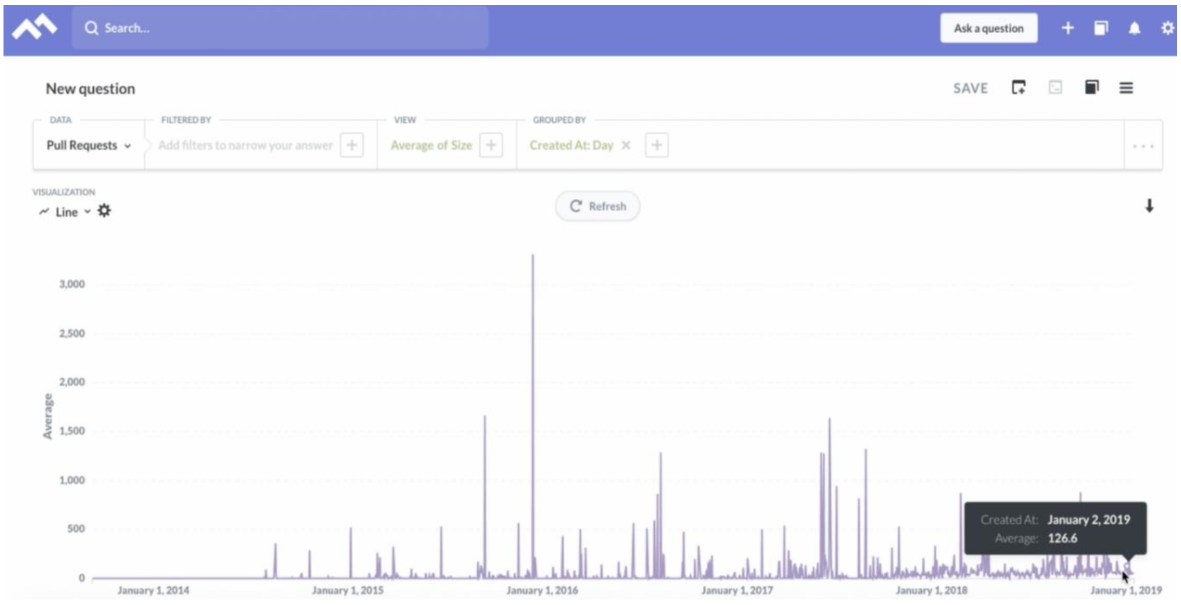


3.2 Velocity

Velocity is a product by the company Code Climate. It helps organisations increase their engineering capacity and productivity by identifying bottlenecks, improving developer experience, and coaching teams with data-driven insights. Velocity is aimed at finding process constraints in departments. They also focus on visualising raw engineering data so that it can be understood by people who may not be software engineers or have experience with software engineering. Using this information, they aim to provide a clear view of progress over time. The goal of Velocity is to answer questions such as:

- Which pull requests are high risk and why?
- Where is my team’s biggest opportunity to improve?
- Are the engineering process changes making a difference?
- Where do developers get held up on projects?

An example of data gathered on Velocity is shown below:[4]



DATA: Pull Requests
FILTERED BY: Add filters to narrow your answer
VIEW: Average of Size
GROUPED BY: Created At: Day
VISUALIZATION: Line

Created At: January 2, 2019
Average: 126.6

3.3 Hackystat

Hackystat is an open source framework for collection, analysis and interpretation of the software development process and data of the product of software engineers. Hackystat allows users to gather data by attaching software ‘sensors’ to their development tools. They feed this information back to the developer or company's server setup. This server provides features to visualise and analyse this data. The engineer may also evaluate work other than software, such as documentation. Hackystat does both client and server-side data collection. It doesn’t make assumptions about connectivity. Hackystat also supports group measurements and can track when multiple engineers are working on the same project.

3.4 Personal Software Process

Personal Software Process (PSP) is a software development process designed to help software engineers improve their own performance by using sound engineering practices. It encourages developers to manage the quality of their own products. It assists in managing software quality throughout the whole development process, evaluating the results of each completed task and using the results to suggest improvements for the next one.

PSP helps software engineers to:

- Improve the quality of their projects.
- Reduce the defects in their projects
- Advance their planning skills.

- Make commitments and schedules they can meet.

Time measurement: PSP recommends that developers should structure the way to spend the time. The developer must measure the time they spend on different activities during the development of a project.

PSP Planning: The engineers should plan the project before developing. Without planning, a high effort may be wasted on unimportant activities which can lead to a poor, unsatisfactory project.

PSP has four levels:

1. PSP 0 – This level includes personal measurement, basic size measures, coding standards.
2. PSP 1 – This level includes the planning of time and scheduling.
3. PSP 2 – This level introduces the personal quality management, design and code reviews.
4. PSP 3 – This final level is for the personal process evolution.[4]

4. Algorithmic Approaches

The Algorithmic Approaches are the ways in which analytics can be performed. These are techniques to assess data sets. The Algorithmic Approach I will discuss is Machine Learning.

Machine Learning

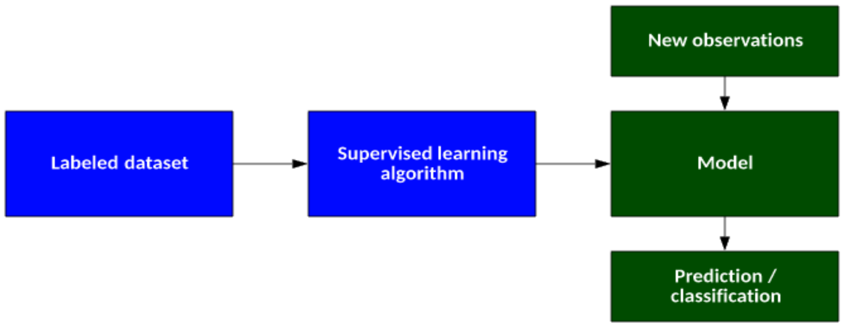
Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.[5]

Machine Learning falls into three primary categories:

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

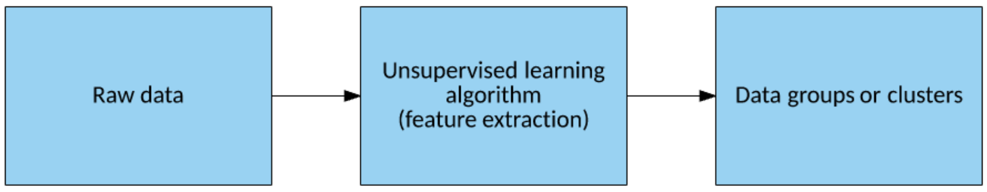
4.1 Supervised Learning

Supervised learning trains itself on a labelled data set. The data is labelled with information that the machine learning model is being built to determine and that may even be classified in ways the model is supposed to classify data.[6] Supervised machine learning requires less training data than other machine learning methods. Training is easier because the results of the model can be compared to actual labelled results.



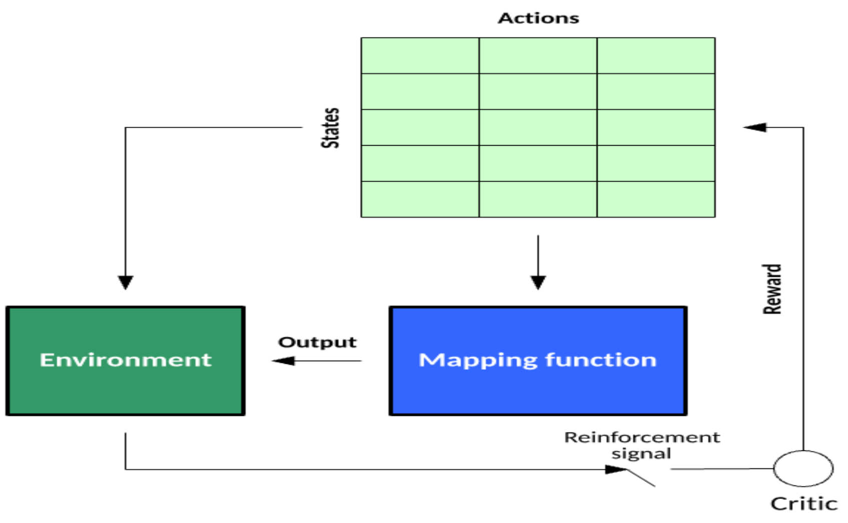
4.2 Unsupervised Learning

Unsupervised learning uses algorithms to extract meaningful features needed to label, sort, and classify the data in real-time, without human intervention.[7] Unsupervised learning is less about automating decisions and predictions and more about finding trends and interactions in data that would be overlooked by humans. You learn more about the raw data that was possibly not apparent through grouping knowledge in unsupervised machine learning.



4.3 Reinforcement Learning

Reinforcement machine learning is a behavioural machine learning model that is similar to supervised learning, but the algorithm isn't trained using sample data.[8] By using trial and error, this model learns as it goes. To create the best recommendation for a given issue, a series of good results will be reinforced. During the learning process, the algorithm randomly explores the state–action pairs within some environment, then in practice of the learned information exploits the state–action pair rewards to choose the best action for a given state that lead to some goal state.



5. Ethical Concerns

Ethics is a system of moral principles. They affect how people make decisions and lead their lives. It is concerned with what is good for individuals and society. Measuring software engineering causes some ethical issues.

As we've discussed above the nature of software development provides the opportunity for a software development company to measure and analyse many aspects of their employee's work life. We've seen earlier how everything from the number lines of code to the number of keystrokes can be captured for every engineer. Let's consider some of the ethical issues with this approach.

- Awareness – are the engineers aware what data is being gathered by the company? It would be a motivation to work faster knowing that one is being monitored but it also causes stress and as mentioned earlier quality may suffer

- Relevance – the data collected must be meaningful and I would question if, for example, the number of keystrokes provides useful info to the overall project outcome
- Accuracy and interpretation – is it possible the data could be inaccurate or open to misinterpretation?
- Use – is the data to be used to increase overall team effectiveness or target poorly-performing individuals for disciplinary measures?
- Privacy – individuals rights to privacy must be respected at all times. There is a legal obligation for companies to do this under General Data Protection Regulation (GDPR)
- Availability – how data is stored and who has access to it is also covered by GDPR. An engineer should be able to obtain the data stored on them. However, does the company plan to publish everyone’s data say in the form of individual performance league tables, for example? This could be dangerous as it could lead to embarrassment for those lower on the table and potential bullying by those higher up.

I believe a software development company should consider all the ethical issues carefully before implementing a measurement system. If done correctly, ethically and accurately I believe the measurement of software engineering is beneficial to individuals and projects and ultimately the business. It can show individuals the areas where they need to improve which may not have been apparent before. It will help a project to identify bottlenecks and improve efficiencies. The business will deliver a better product in better time scales

6. Conclusion

Software companies need to deliver software that meets the agreed specifications on time and within budget. The goal of using software metrics is to use data to drive improvements in software development quality and productivity. Companies who do not use data-driven software development techniques effectively may struggle to meet project deadlines and may not survive in the long term.

7. References

1. https://www.infopulse.com/blog/top-10-software-development-metrics-to-measure-productivity/
2. https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/Web/Planning/productivity.html
3. https://codeclimate.com/blog/velocity-vs-gitprime/
4. https://codeclimate.com/blog/velocity-launch/
5. https://www.geeksforgeeks.org/personal-software-process-psp/
6. https://www.expert.ai/blog/machine-learning-definition/
7. https://developer.ibm.com/articles/cc-supervised-learning-models/
8. https://developer.ibm.com/articles/cc-unsupervised-learning-data-classification/
9. https://developer.ibm.com/articles/cc-models-machine-learning/#reinforcement-learning

8. Further Reading

http://www.nextlearning.nl/wp-content/uploads/sites/11/2015/02/McKinsey-on-Impact-social-technologies.pdf

https://www.researchgate.net/profile/Jan_Sauermann2/publication/285356496_Network_Effects_on_Worker_Productivity/links/565d91c508ae4988a7bc7397.pdf