MEASURING SOFTWARE ENGINEERING

About me

Student Number : 17329137

Name : Holly Baker

Course : Computer Science

Contents

Introduction

Software engineering is the systematic application of engineering approaches to the development of software. (En.wikipedia.org, 2019) In this industry, there is little standardization of software metrics, therefore there is an array of metrics which are analyzed to assess the software engineering process. Each metrics's impact is positive in one particular situation and negative in another.
With this, there are many computational platforms which offer an engaging way to track software development and measure performance. These platforms make use of these different metrics and algorithms, and present it in a visualizing way for the user. The choice of the metric or algorithm is made by the leading body in the organization, this choice is an important ethical dilemma.
Within this framework, the four sections that will be discussed in this report are measurable data, computational platforms, algorithmic approaches and ethics.

Measurable Data

Introduction

There is no universal measure of workers productivity across different occupations. However there are various measures that can capture a software engineers performance in their specific settings.
When the right performance measures are assessed, the following benefits emerge:

- ❏ It provides detailed information about the software engineer's productivity to employees.
- ❏ It helps answer important questions e.g which incentives work.
- ❏ It enables the design of appropriate contracts.
- ❏ It improves productivity in the workplace.

In contrast to these advantages:

- ❏ It's impossible to measure all multidimensional aspects of the engineer's productivity.
- ❏ Wrong measures can lead to generating the wrong results.
- ❏ And in relation to observing at a team level, it's not always possible to estimate individual contributions.(Sauermann, 2016)

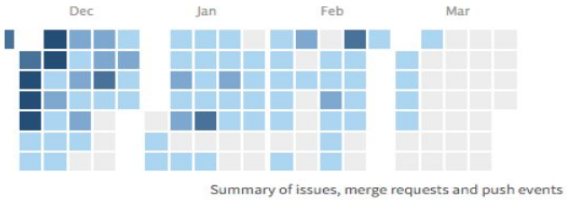
Overview of what can be measured

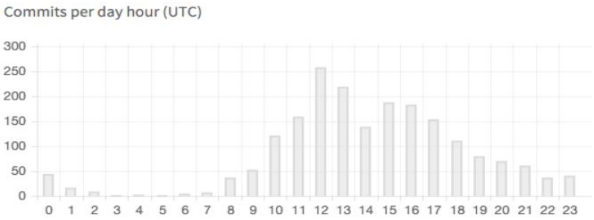
When monitoring all workers over all industries, there is a lack of reliable methods to determine the universal productivity of workers. However the computer science industry holds onto a handful of tools which assess performance measures.
One important tool is Github, the American company that provides hosting for software development version control using Git. In May 2019, Github report of having over 37 million users and more than 100 million repositories, making it the largest host of source. (En.wikipedia.org, 2019).

Any respectable software engineer is an owner of a GitHub account. The website is often used as a source when hiring employees, showing all the candidates previous projects.
This tool carries a bundle of accessories that measure performance, for example, the amount of commits made or the number of lines of code (LOC) in a project. However these factors of calculating the employees work can be seen to be the least significant as they can filter out other valuable work that might be untraceable.
There are more empathetic measures that value employees, such as asking coworkers to rate their helpfulness in the workplace or asking the employee of interest to show evidence of any unshown contributions. (GitLab, 2019)
Alongside this other measurable data can be extracted from metric programs or working hours of the employee.

Commits

Commits are arbitrary changes captured in a single moment in time. They can be viewed as an indication of activity and technically they are a measure of performance. However their size and frequency do not correlate with the work needed to achieve that change. Commits do not equal productivity. Instead, the consistent progress of committing can be more honest in measuring performance.
The definition of progress, is someone who takes deliberate steps to work towards solving the problem they are working on. Software engineers communication and activity through commits is an efficient way to gather measurable data. As they make commits to further their progress and communicate about their work.
GitLab presents the following diagrams , which show the activity of commits on a profile and also how everyone in a project is committing on average during the day:





In addition to this in GitLab Enterprise Edition, you can get a better detailed overview of the activity of everyone in a team through the Contribution Analytics:

Contributions per group member

Name	Pushed	Opened issues	Closed issues	Opened MR	Accepted MR	Total Contributions
Robert Speicher	0	0	0	0	0	0
Marcia Ramos	2	2	6	0	0	25
Kamil Trzciński	0	0	0	0	0	0
Achilleas Pipinellis	26	1	24	2	8	107

However these only give aid to measuring performance, they do not calculate it. By combining this, their progress and more data in smart ways, there is a better insight into a developers activity.(GitLab, 2019)

LOC

The concept of judging performance on a number of commits is similar to examining a developer's ability on the number of lines of code, both can be irrational yet viable. It is a traditional inadequate model from the 1960s,where it was used routinely.

However ,LOC is simply to understand and easy to collect, many present industries have accepted this form of measure.

The amount of non-comment code lines can correlate with:

- ➔ Undetected errors
- ➔ Maintenance costs
- ➔ Training time for new contributors
- ➔ Migration costs
- ➔ New feature costs
- ➔ Cost of optimization etc. (Medium, 2019)

It has been a success as it is easy to compute and can be visualised. For example employers understand wat 100 LOC program looks like.

It is still routinely being used in its original application: as a normalising measure for software quality.

Metrics programs

In the 1970s, LOC drawbacks with the aspects complexity,functionality and effort were recognised. Alongside the increased diversity of more programming languages, the industry acquired an explosive interest in measures of software complexity.

The industry began to try develop successful Metric Based management decision support tools .

The most successful example is : Bayesian belief nets (BBNs)

BBNs has attracted attention for the solution for problems of decision support under uncertainty. BBNs underlie the help wizard in Microsoft Office and has been used in the area of software assessment. (E. Fenton and Neil, 2019)

The idea of working on the mechanics of implementing metrics programs in order to measure performance is logical and attractive. However industrial metrics activity is poorly motivated. The metrics being practised are essentially the same basic metrics that were around in the late 1960s, LOC. (E. Fenton and Neil, 2019)

Working hours

Another measure of data, is the number of hours worked. However, working hours can affect productivity,such that when hours worked increase, the agents become less productive, as fatigue can play an important role. Therefore hours worked can be a measurable data, but employees must keep in mind the effect on the workers productivity.(J. Lindquist, 2015)

Conclusion

To conclude, there is no universal definition of worker productivity, it depends on the settings. In a computer science setting, the software engineer's output can be measured through Commits, LOC, Metric programs and working hours . All measures noted have a deficit of accuracy, for example, an engineer has made multiple commits however the quality of these commits are very poor. Many factors affect productivity such as time driven measures or rewarded incentives. Choosing the right performance measures is a must as it informs and improves the decision-making in the management and policies affecting the workers. An employer's nature will determine which measure is ethically and morally correct to apply.

Computational Platforms

Introduction

There are many analytic applications that are used to turn large amounts of data into meaningful insights. Examples of these computational platforms which are available to measure and assess a software engineers process are Gitprime, Hackystat and CodeClimate. Each bring a distinct set of strengths and its own measurement philosophy. (Gitclear.com, 2019)

GitPrime

GitPrime is an organizational tool, pioneering a different way of measuring and communicating about productivity in software engineering.It provides engineering leaders with metrics in context to ask better questions and advocate for the team with substantive data:

- Did the change to our sprint cycle have a net-positive effect?
- How much of your team's burn is going to refactoring old code? (GitPrime Help Center, 2019)

The architecture of the application is framed by four key metrics: "active days," "commits per day," "impact, " and "efficiency." It bends LOC into useful data.

Gitprime's case studies include a 137% increase in Impact by Storyblocks, a 40% decrease in bugs introduced from Cloudhealth and a 25% increase in measured Impact enjoyed by Adext. (Gitclear.com, 2019)

It breaks away from basic inadequate metrics but instead assemblies these metrics together in a sheer quantity of different reports offered.

Hackystat

Hackystat is another open source framework for automated collection and analysis of software engineering process and product data.(Hackystat, 2019) The service-oriented architecture is completely based on open-source components and standard protocols such as XML. (Sillitti, 2019)
It holds a competitive advantage as it allows engineers to use any language. Meaning, developers can build native client user interfaces that interface to web-based Hackystat services, allowing them to explore advanced 3D visualizations. (M Johnson, 2019). However service-oriented architecture is not cheap, it carries concrete performance costs and it also creates new kinds of coordination issues.

CodeClimate

Code Climates mission is to provide meaningful and actionable engineering insights for the entire engineering organization. (Codeclimate.com, 2019). Their key product they offer is, Velocity, which analyzes all the data from a developers GitHub repos and provides you with head-up-displays, real-time analytics, and custom reports.
Both GitPrime and Velocity share a common thread of ideology and features. However in contrast to GitPrime, Velocity draws attention to actual artifacts of work that could be problematic, where as GitPrime draws attention to the problematic contributors. *(Gitclear.com, 2019)*

Conclusion

Theses developer measurement companies, GitPrime, Hackystat and Code Climate, have increased the ability of managers to assess their workforce. There is still a large opportunity for smarter companies to recognize that they can gain an information advantage in engineering and with this, the tools will keep growing and become more polished. *(Gitclear.com, 2019)*

Algorithmic Approaches

Introduction

To characterise a developers performance different algorithmic approaches are used. An algorithm analysis is a form of a software quality metric alongside LOC . The following techniques Halstead Software metric, Cyclomatic Complexity Number metric and Function Points Analysis share the common goal of measuring the quality of software.

Halstead’s Software Metrics

Halstead’s metrics are included in a number of current commercial tools that count software lines of code.(GeeksforGeeks, 2019)
The algorithm counts tokens, then determines which are operators or operands.The size of code is the sum N of the number of operators N1 and operands N2. It is highly correlated to the number of LOC,

with the improvement that more complicated lines will contribute more to the metric than simple lines. (Nickerson.to, 2019)

By identifying and counting these two types of attributes we can define the following metrics:

- ➔ Length,
- ➔ Vocabulary,
- ➔ Volume,
- ➔ Level,Difficulty,level estimator and
- ➔ Intelligence content.(Abran, 2019)
- ➔ Programming effort
- ➔ Programming Time

Each metric has their own equation and derivation.

The program length formula, is the total of distinct, unique operators and operands:

$$\hat{N} = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$$

where η_1 = the number of distinct operators

η_2 = the number of distinct operands

The formula for Volume, is the length of a program multiple by the minimum number of bits necessary to represent the operators and operands:

$$V = N \log_2 \eta$$

where $N = N_1 + N_2$

$\eta = \eta_1 + \eta_2$

With this intel, Halstead builds specific characteristics of a program, which enables managers to visualize a developers process and performance.

Halstead is favoured as it does not require in-depth analysis of a programming structure. It can measure overall quality of programs and it can predict programming effort, the rate of error and maintenance.

In contrast to these benefits, difficulties arise when trying to differentiate operands and operators. And the model is not capable of measuring the structure of a code inheritance, interactions between modules.(De Silva, 2019) However , it does use token count which is more accurate than LOC, as it deals with the complexity of lines.

Cyclomatic Complexity Number Metric

The most widely used graph metrics is cyclomatic complexity, defined by Thomas McCabe. It is based on the control flow structure of a program, unlike Hastlead deriving results from equations.

The procedure's statements are transformed into a graph. Then the cyclomatic complexity is calculated by measuring the linearly independent path in the graph and is represented by a single number.

The metric is defined in two equivalent ways:

(1) the number of decision statements in a program + 1

(2) for a graph G with n vertices, e edges, and p connected components,

$$v(G) = e - n + 2p$$

This complexity measure is simple and present in most existing tools. However McCabe's scheme does not acknowledge what is inside a conditional statement, which increases the possibility of results being inaccurate as it could miss a complex expression that should be calculated.
This metric only takes into account the contribution factor control flow complexity, which leads to an incomplete view when calculating the complexity of a program.

Function Point Analysis

Another algorithm approach is Automated Function Point Analysis. It utilizes a sophisticated algorithm designed to streamline the counting.Evaluating software quality, complexity, risk and team productivity by using a function point scoring system. (Castsoftware.com, 2019)
Function points (FP) measure software size accurately. As sizing is clearly an important component in determining a software's productivity. (Softwaremetrics.com, 2019)
The Function point metric is a weighted sum:

```
function points =
        the number of inputs * 4 +
        the number of outputs * 5 +
        the number of inquiries * 4 +
        the number of master files *10
```

Function Point Analysis is useful as it eliminates the need to evaluate each line of code. It can estimate the overall rate of progress in software productivity and enabling software quality to improve.
It is also useful as it can be used to determine whether a tool, a language, an environment, is more productive when compared to others.

Conclusion

The estimation of the size of software is key for assessing a software engineers process and productivity. These algorithmic approaches capture the size of software but also the complexity which allow managers to further predict the effort and time which will be needed to build the project.
(GeeksforGeeks, 2019)

Ethics

Introduction

Ethical questions arise when measuring software processes. There are many aspects that should be considered in the decision-making around a metric, such as the business priorities or the cost of the metric.

The shape of the working environment will change depending on the analytic chosen to assess the employee's performance. Certain analytics, in particular Lines of Code, can create an unpopular working environment for developers due to the unfair metrics and will result in driving valuable talent out of the company.

Choosing metric's

A toxic engineering culture can be created when metrics are used in the wrong way. For this reason employees need to be careful about which metrics to track and how to track them. (Hackernoon.com, 2019) Ethical questions arise in the decision making process of choosing metrics. The following points below are important to observe before establishing metrics:

- Don't' use software metrics that do not lead to change.
- Link software metrics to business priorities
- Metrics should be easily understandable.

Business Priorities

A company must link their software metrics to their business priorities. (Hackernoon.com, 2019) They must select the ones which are most relevant to their current business priorities.

In relation to this, ethical concerns arise as software managers have the decision to either act out of self-interest or to accept their moral responsibilities. (Kerssens-van Drongelen, 2019) They carry the responsibility of the employees welfare, as dismissing ethical metrics for corrupt one's due to wanting the minimum cost , can lead to a stressful working lifestyle for employee's.

Companies who encourage the personal growth of their workforce and value motivating employees , will want to develop a moral set of metrics. However Companies who gain success through acting greedy and having a high turnover, will favour unethical metrics.

Following this poorly chosen metrics, will create employees with bad habits. For example, employees with bad code can be skewed by making LOC a performance metric.

Concerns around LOC

The assessment of LOC, can be provided at very little cost, making it a popular metric. However LOC can be considered to be a "professional malpractice".(Mas y Parareda, 2019) It is unfair, as it discredits a developers contributions as there is a lack of correlation with functionality and effort.

The employees valuable contributions is not reflected in this metric.

Therefore it can be a good metric for measuring velocity for example, but a poor metric for measuring an individual productivity for raises.

Conclusion

When teams try to keep track of their improvements through a set of chosen indicators, the leaders morality determines the set of indicators the team will follow. Thus, the leader's ethical stance is key to the organization.

Conclusion

The report above is divided into 4 sections, which are measurable data, computational platforms, algorithmic approaches and ethics. Each section illustrates contrasting elements and features on how to measure and assess the software engineering process. Due to the lack of clarity and standardization on this theme, There is a multitude of diverse viewpoints. During the whole report, my judgement on the issue is rooted in the way the report is delivered.

BILBROGRAPHY

- *En.wikipedia.org. (2019). Software engineering. [online] Available at: https://en.wikipedia.org/wiki/Software_engineering [Accessed 4 Nov. 2019].*
- *Sauermann, J. (2016). Performance measures and worker productivity. [online] Available at: https://www.researchgate.net/publication/303365666_Performance_measures_and_worker_productivity [Accessed 28 Oct. 2019].*
- *En.wikipedia.org. (2019). GitHub. [online] Available at: https://en.wikipedia.org/wiki/GitHub [Accessed 28 Oct. 2019].*
- *GitLab. (2019). Commits Do Not Equal Productivity. [online] Available at: https://about.gitlab.com/blog/2016/03/08/commits-do-not-equal-productivity/ [Accessed 28 Oct. 2019].*
- *Medium. (2019). Assessing Employee Performance. [online] Available at: https://medium.com/javascript-scene/assessing-employee-performance-1a8bdee45c1a [Accessed 28 Oct. 2019].*
- *[closed], H., Cross, B. and Tillemans, P. (2019). How bad is SLOC (source lines of code) as a metric?. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/3769716/how-bad-is-sloc-source-lines-of-code-as-a-metric [Accessed 28 Oct. 2019].*
- *E. Fenton, N. and Neil, M. (2019). [online] Citeseerx.ist.psu.edu. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.63.2683&rep=rep1&type=pdf [Accessed 28 Oct. 2019].*
- *J. Lindquist, M. (2015). Network Effects on Worker Productivity. [online] Available at: https://www.researchgate.net/profile/Jan_Sauermann2/publication/285356496_Network_Effects_on_Worker_Productivity/links/565d91c508ae4988a7bc7397.pdf [Accessed 28 Oct. 2019].*
- *GitPrime Help Center. (2019). What Is GitPrime Exactly?. [online] Available at: https://help.gitprime.com/general/what-is-gitprime-exactly [Accessed 28 Oct. 2019].*
- *Hackystat. (2019). Hackystat. [online] Available at: https://hackystat.github.io/ [Accessed 28 Oct. 2019].*

- *M. Johnson, P. (2019). [online] S3.amazonaws.com. Available at: https://s3.amazonaws.com/academia.edu.documents/42851775/Beyond_the_Personal_Software_Process_Met20160219-28245-1ijy2pj.pdf?response-content-disposition=inline%3B%20filename%3DBeyond_the_Personal_Software_Process_Met.pdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWOWYYGZ2Y53UL3A%2F20191028%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20191028T161622Z&X-Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-Signature=caad104098cac0f191f17ab6581418718bb1bc3ba1686f9f8411cd51da8b9dc6 [Accessed 28 Oct. 2019].*
- *Sillitti, A. (2019). [online] Citeseerx.ist.psu.edu. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.138.6806&rep=rep1&type=pdf [Accessed 28 Oct. 2019].*
- *Gitclear.com. (2019). Measuring Developer Productivity A Comprehensive Guide For The Data Driven - GitClear. [online] Available at: https://www.gitclear.com/measuring_developer_productivity_a_comprehensive_guide_for_the_data_driven [Accessed 28 Oct. 2019].*
- *M Johnson, P. (2019). [online] Available at: https://www.researchgate.net/publication/228731372_Experiences_with_hackystat_as_a_service-oriented_architecture [Accessed 29 Oct. 2019].*
- *Medium. (2019). Assessing Employee Performance. [online] Available at: https://medium.com/javascript-scene/assessing-employee-performance-1a8bdee45c1a [Accessed 28 Oct. 2019].*
- *GeeksforGeeks. (2019). Software Engineering | Halstead's Software Metrics - GeeksforGeeks. [online] Available at: https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/ [Accessed 30 Oct. 2019].*
- *Abran, A. (2019). Software Metrics and Software Metrology. [online] Available at: https://www.academia.edu/16848984/Halstead_Metrics_Analysis_of_their_Design [Accessed 30 Oct. 2019].*
- *Nickerson.to. (2019). LIMITS OF THE VISUAL. [online] Available at: http://www.nickerson.to/visprog/ch7/ch7.htm [Accessed 30 Oct. 2019].*
- *De Silva, T. (2019). On the Improvement of Cyclomatic Complexity Metric. [online] Available at: https://www.academia.edu/9055656/On_the_Improvement_of_Cyclomatic_Complexity_Metric [Accessed 30 Oct. 2019].*
- *Softwaremetrics.com. (2019). Fundamentals & Introduction of Function Point Analysis. [online] Available at: https://www.softwaremetrics.com/fpafund.htm [Accessed 30 Oct. 2019].*
- *Castsoftware.com. (2019). Function Point What Is It, Info, Video and Free WHITEPAPER. [online] Available at: https://www.castsoftware.com/glossary/function-point [Accessed 31 Oct. 2019].*
- *GeeksforGeeks. (2019). Software Engineering | Project size estimation techniques - GeeksforGeeks. [online] Available at: https://www.geeksforgeeks.org/software-engineering-project-size-estimation-techniques/ [Accessed 31 Oct. 2019].*
- *InfoQ. (2019). Opinions: Measuring Programmers' Productivity. [online] Available at: https://www.infoq.com/news/2008/10/measure-programmers-productivity/ [Accessed 1 Nov. 2019].*
- *Mas y Parareda, B. (2019). Measuring Productivity Using the Infamous Lines of Code Metric. [online] Itestra.com. Available at: http://itestra.com/wp-content/uploads/2017/08/07_itestra_measuring_productivity_using_lines_of_code.pdf [Accessed 1 Nov. 2019].*
- *Hackernoon.com. (2019). 7 Rules to Track Software Engineering Metrics Correctly. [online] Available at: https://hackernoon.com/how-to-use-and-not-abuse-software-engineering-metrics-3i11530tr [Accessed 4 Nov. 2019].*
- *Kerssens-van Drongelen, I. (2019). Ethical Dilemmas in Performance Measurement. [online] Available at: https://www.jstor.org/stable/pdf/25075055.pdf?refreqid=excelsior%3Aed8301e624e4b7bb7b48c4e4afa1635b [Accessed 4 Nov. 2019].*