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

School of Computer Science and Statistics

Assessment Submission Form

Student Name	Patrick Nulty
Student ID Number	16318849
Course Title	MSISS
Module Title	Software Engineering
Lecturer(s)	Stephen Barrett
Assessment Title	Measuring Engineering
Date Submitted	20-11-2018
Word Count	3646

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at: http://www.tcd.ie/calendar
I have also completed the Online Tutorial on avoiding plagiarism 'Ready, Steady, Write', located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write
I declare that the assignment being submitted represents my own work and has not been taken from the work of others save where appropriately referenced in the body of the assignment.

Signed Patrick Nulty Date .....

Introduction

Software Engineering is a discipline of engineering which is a method of design, development, management and sustaining software. Originating in the 1960s it is a relatively young but is rapidly growing and has come on remarkably in that time. The constant struggle of having to maintain and improve software in other to keep it working efficiently and reliably for our needs.

In this report, I will discuss the following four sections:

1. The ways in which the Software Engineering process can be measured and accessed in terms of measurable data.
2. An overview of the computational platforms available to perform work of this kind.
3. Algorithmic approaches available.
4. Ethics concerning this kind of analytics.
5. Conclusion.

Measurable data

"Without data
you're just another person
with an opinion."

W. Edwards Deming

When analysing the means in which a software engineering process is evaluated and measured, data becomes essential. Data is very dependent on its careful collection, which is easier said than done. Originally it was much more difficult to get any data to measure. Developers would be more focused on their code rather than spend hours on a practice that they couldn't comprehend any use for at the time. It was also extremely inconvenient for a developer to measure this data as it had to be inputted individually on a spreadsheet with analysis being done by hand after this. This was deemed to be too expensive and often mistakes were made due to human error. For this reason, data was difficult to find in the 1960's and 1970's and this slowed down the development of Software Engineering.

Quality

It is often the case that an organisation would see data quality as a given and that it just simply exist. However, this is not the case. It is important for an organisation to set out parameters that clearly define what it can include, so 'good data' can be achieved and remains reliable. The main factors that affect data quality are:

Relevance: Data must be set within a time period, place and/or population that effect and be directly related to what is being analysed. It is often the case that you will spot that the data is not relevant.
Accuracy: The data must actually be correct and must always stay relevant to the analysis that you are performing. Coverage must represent the whole population in order for the analysis to be accepted.
Completeness: The dataset should describe all appropriate aspects of the problem that is being assessed. Most importantly is that there is no absent data within the area, time period and/or population that the data is interpreting.
Timeliness: Quality data should be recent to the business. As data is always in motion it is optimal to get the most up-to-date version of the data as it can be the most valuable part of the data set.
Cleanliness: Data should be prepared, standardized, structured and named or documented to the extent achievable. It must also be free of duplicates and look well. It is easier to spot clean data as it will look presentable.

If this is achieved, a dataset will include data of good quality. If not, then generate or acquire new data.

Data is continuously in motion. In some instance data changes by the second. Each action made has an impact on the quality of the data. Often 'good data' can become 'bad data' quite suddenly if it now has irregularities in the data set. It is said that data quality is not obvious and it can be eroded by various circumstances. For this reason, businesses spend a lot of money actively guaranteeing that their data is of good quality.

Reviewing Data

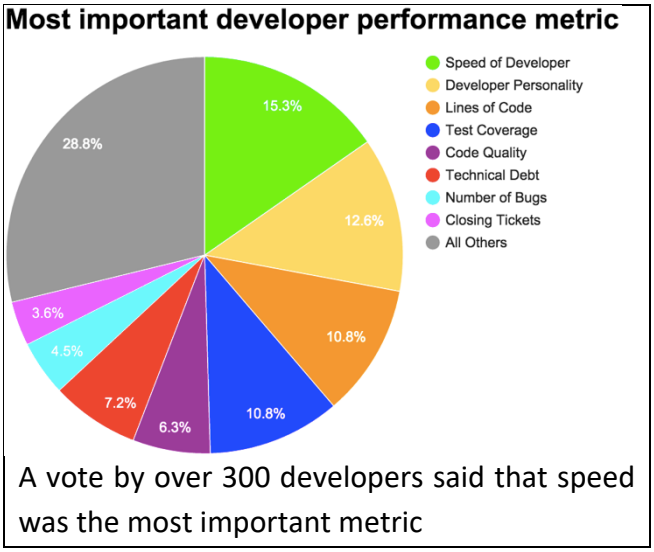
The following headings are the most important software metrics when measuring and accessing the performance of a software engineer.

Number of commits: This is a good indicator that will show the activity of the programmer as it shows the number of times he or she has contributed to the code. A good software engineer will not write countless lines of code but will consistently try to improve it by either adding or removing code. This may not show the quality of the code but it gives an indication of an engineer’s work-rate. This is frequently monitored on sites such as 'Git' by managers to see which employees are working hard.

- **Code churn** represents the number of lines of code that were modified, added or deleted in a specific period of time. If the code churn increases, then it might indicate that the software development project is in need of attention. This gives us a good insight into the quality of the code. It's frequently the case that the more lines of code

will cause more defects. Spikes in code churn can help visualize where a problem has arisen.

Speed: As with all jobs, speed is crucial. This metric can be broken down into two parts. Cycle time and Velocity. Cycle time refers to the time period in which it takes to implement changes to the software system. Whereas, developers velocity is measuring the number of software units it completes in an iteration. This is measured over a set time scale, usually how long it would take to finish a project. Detailed time logs must be taken to ensure members are not falling behind the rest of the team.



A vote by over 300 developers said that speed was the most important metric

Bug fixing: This interesting metric shows how much time that a software engineer spends over a week/month fixing bugs. The least error (shown in the below formula) in the test will mean a greater efficiency from a software engineer. A software engineer with a least bugs will have least maintenance to perform in the long run. Although it is said that a developer who fixes a lot of bugs is said to be effective. However, if he/she makes all of these mistakes in the first place, it can slow down productivity.

$$\left(\frac{Bugs\ found\ in\ Test}{Total\ bugs\ found(bugs\ found\ in\ test\ +\ bugs\ found\ after\ shipping}\right)\times 100$$

Technical debt: If a software developer chooses an easier type of solution and does not think of the long-run. It may be less effective and cost the team valuable time in the future, instead of finding the optimal solution. The cost that is now implied is called technical debt. It can arise due to the laziness of a developer and is tried to be avoided at all times. This metric can be represented as a percentage. I would say it gives the greatest insight into how a software developer operates.

Coverage: This metric (usually a percentage) is the amount of code that a test would include. This is a key component for a software developer as it will test that their code is robust under different conditions. By thoroughly checking their code a developer will have a greater solution and a higher test accuracy.

A software developer should not only be measured by his/her ability to code but should also be measured on other factors such as emails, meetings attended and communications such as time on the phone. Although it may be considered to be intrusive if a company is to monitor

these factors but this data is extremely useful for getting an indication of the productivity of the developer.

Summarizing, measuring data is an extremely useful factor when trying to understand more about the developer. The challenges lie in the generation of the data and doing so in an honest and respectful manner.

Where to compute?

Following on from measurable data now we will look into how you would go about computing this data. Ideally, we want all the available data so we can use it to create graphs, plots and stats that will provide us with a better idea of how we can improve the processes of a software engineer. But unfortunately, this is not always the case as we must compute the raw data first in order to draw any of these conclusions.

Development of Computational methods

Personal Software Process: (PSP) is a structured software development process that is intended to help software engineers understand and improve their performances by using a data-driven procedure. Although this idea was to benefit the developer, the forms in which a developer would fill in were incredibly time-consuming as they would have to be filled in by hand and not entirely accurate as a developer would have to fill in their own form in cases causing human error.

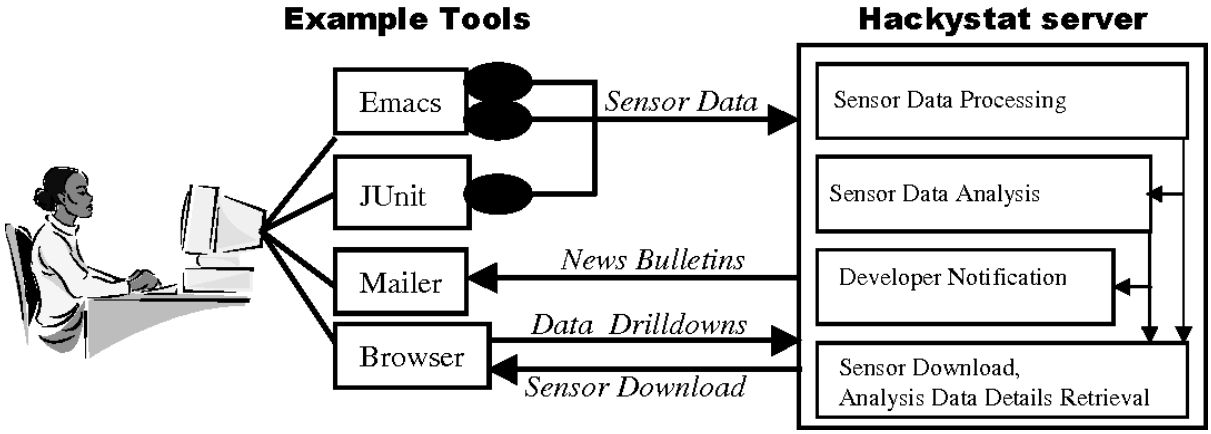
Although it was considered a good concept it had many other problems. It wasn't good practice as it wasn't flexible and it was too time-consuming. For these reasons, it was not universally accepted but it started a change up as the mistakes in it were fixable.

Leap toolkit: When PSP didn't take off it made an opportunity for the toolkit called 'Leap'. The idea was that it wanted to overcome the data quality problems that were associated with

PSP by automating and normalizing data analysis. It now automated the data analysis part but the developer would still have to enter most of the data by hand but now. It is portable and maintains the data about the individual developer's activities without referencing to his/her name in the file. Hackystat was the data collection tool used to analyse the data that was collected.

Hackystat

Developed in the University of Hawaii, Hackystat was the third generation of PSP data collection. It is a framework that enables developers to collect and analyse PSP data automatically, sensors would send the information to a centralized system using the SOAP protocol. Hackystat focuses on individual data collection and the use of this data while now giving more privacy for the developer. It also had the ability to track any interplay between the developers, showing who made the previous edits to the code which was very useful in showing how a team operated.



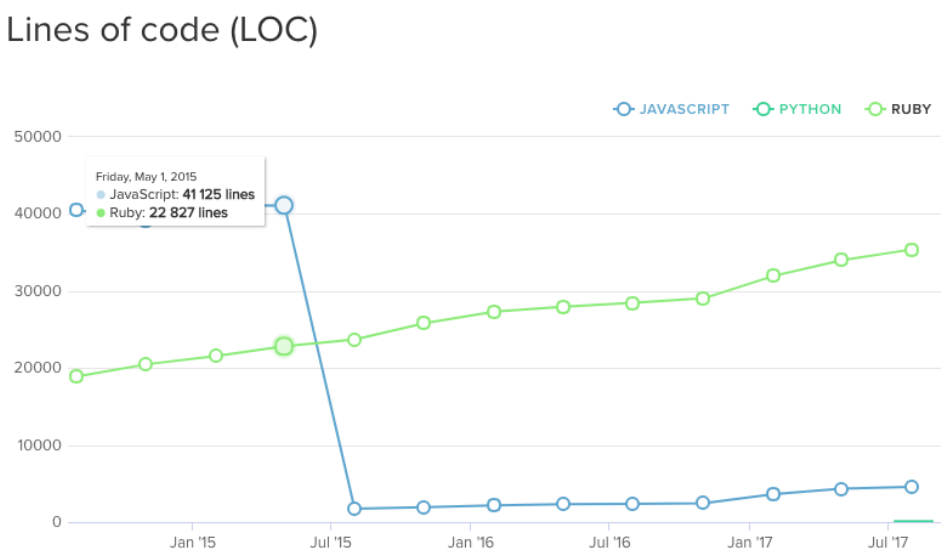
Modern Computational methods

Presently the market for measuring software engineering has significantly increased. The field has transformed by the advance in the industry and now there is a whole new market with companies providing platforms with various different features to compute data to output it in a user-friendly way. As different businesses have different requirements, some tools that work well for some businesses might not suit another. The providers of these tools that are currently leading the market are Codebeat, Codacy, Code Climate and Scrutinizer only to name a few.

Code Climate

This is a well-developed and well-built tool that can be used in a number of situations. It incorporates fully configurable test coverage and maintainability data throughout the development workflow. It is also the choice of many of the top players in the industry such as Pivotal and New Relic. Every day the tool analyses over 2 billion lines of code in a day and is

used in more than 100,000 projects. It also displays the data in easy to read trend graphs making it more understandable to anyone looking at the output.

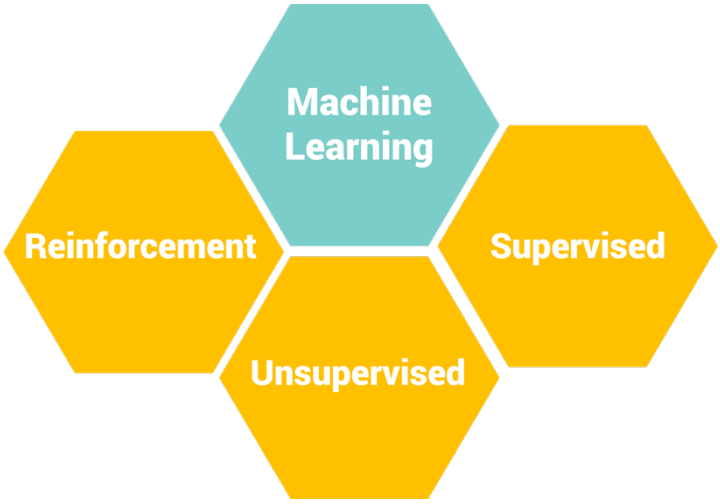


Codacy

Founded in 2012, it is a relatively new company and has succeeded so far for reasons such as it having a superior user interface than its competitors. Like Code Climate, it also is very flexible and has various different features. It has acquired big names such as PayPal and Adobe. A young company that is growing quickly and has much potential. It is a good example of how the industry is rapidly growing as it has over 65,000 developers.

These various types of approach are similar in idea but have differences in the way they apply certain features of its structure. For this reason, one way may be more ideal for some companies whereas a different company might not think that this computational method is right for them. In the interest of the metrics of a developers behaviour and their practices then it is best to use a modern computational method although the may not give as much depth in the analysis, it does give a less chance of human error in data collection.

Algorithms



Machine learnings or Computational Intelligence (CI) are a big part of how we now measure data. CI is the ability to work with incomplete and unclear information like data. Its concept to how it works is similar to how the human brain operates. The idea is that machines can 'learn' to make predictions and identify patterns without being explicitly programmed to do so, using data-driven methods. Machine learning can be broken up into three different parts: supervised learning, unsupervised learning and reinforcement learning. Now I will go into detail about them.

Supervised Learning

This is where you have a set of data and you teach your algorithm to perform classification and regression techniques in order to learn the mapping function from the input to the output, where the output is known before the data runs through the algorithm. This is used in the majority of practical machine learning. An accurate way of describing this in more detail is a student (the input data) doing a course set by an instructor (the output analysis). The student gains information (data) by sitting the course. You get a great in-depth detail of your classes from supervised learning, meaning that you can make a perfect decision boundary to distinguish different classes accurately.

Depending on the type of data that will be given, it will either have to use a regression or a classification technique. A **random decision forest** can be used in both situations. How they work is by building multiple regression trees and merging them together, a more stable and accurate solution is obtained.

Unsupervised Learning

This is when you have an input of your data and there is no output (teacher in the last scenario) or right or wrong answer. By modelling the underlying structure or distribution in the data, we learn more about the data. Algorithms are left to their own devices to discover and present the interesting structure in the data. Unsupervised learning problems can be further grouped into clustering and association problems. It is often the case that the analysis result is due to the interpretation of the results by whoever preforms the analysis. K-means clustering and Principle Component Analysis are methods of unsupervised learning

Principle Component Analysis

PCA is a data-driven procedure that performs an eigen decomposition on a covariance matrix. It is a dimension reduction technique used to find similarities within a set of data and is used to describe a set of correlated variables in terms of a new set of uncorrelated variables. We find linear combinations of the original data to find maximum variance. By re-expressing the data so it reveals its internal structure and it explains its variation through the use of a linear combination from the original variables.

Reinforcement learning

The final type of machine learning allows machines and software agents to automatically determine the ideal behaviour within a chosen data set in order to maximize performance. A reinforcement signal is reward feedback that is given to the agent so it can learn the behaviour of the data. Meaning that less time will be spent on creating a solution. MDP (Markov Decision Process) is an example of this algorithm.

Markov Decision Process

MDS attempts to find a map in a lower dimension so that inter-point dissimilarities match those in the original dissimilarity matrix as much as possible. MDS can be performed in a few ways. In the metric case, the lower dimensional distances arise as a result of applying a continuous monotonic function to the distances in the original dimensions. The results are the same between classical metric scaling with Euclidean distance and PCA. In the non-metric version, the order of the rank of the distances needs to be preserved. This algorithm is best described in the context of a maze. The system moves through the maze systematically learning and recognizing the fastest possible route out of the maze.

Although these algorithms are not always completely accurate, they save a lot of time and effort. The advantages far outweigh the disadvantages and by eliminating human error the results are generally more accurate.

Ethics

The questions that I want to find out more about are, is this morally right or wrong to share your private information with an organisation and for them to use it for their advantage? And is it really your data once you give it to this organisation?

In today's world, we hear a lot about data privacy, but there seem to be no boundaries at the moment. You are being monitored every time you type in your email address and bank details, which for most people is a countless number of times a month. You entrust this data with various companies who store your username, email, passwords and credit card information, information that could be used to steal from you and/or even incriminate you.

In 2013, LinkedIn a well-renowned company were caught hacking into users accounts. From the users account, they sent spam emails to anyone who was on the mailing list of that user and proceeded to email these account with three separate invitations to join their LinkedIn network. In September of 2013, a court case was brought against them which the company ended up paying out $13 million in a settlement.

GDPR

In May 2018, GDPR (General Data Protection Regulation) has come into effect by European law. The law basically observes all organisations within the EU including those who sell goods and services within the EU. It also processes the personal data of EU residents. It essentially covers four actions that companies had to comply with:

1. **Personal Data**: An organization must encrypt and pseudonymize personal data.
2. **Security Testing**: Regular testing and assessments must be completed to show the effectiveness of the security systems.
3. **Provisions**: Provisions for integrity, confidentiality, availability and resilience of processing systems and services.
4. **Physical or technical incident**: In this event, organisations are authorised to restore the availability and access to personal data after a period of time.

The punishments are severe as the GDPR authorities can issue fines up to €20 million or 4% of their annual worldwide turnover. Although it may seem like a hindrance for companies, it definitely gives people a greater sense of security.

Boundary's

We assume that when we provide data, it is being used for an honest reason, but there is more to it that most people can understand. Where is the line drawn between 'my data' and the data that is available for analysis? As mentioned before an employer can easily monitor

an employee through emails, phone calls and the time that they spend online. Modern technology has gone a step further, there are now new ID badges that allow employers to listen to everything you say, see how much time you spend with people or at your desk. An employee can opt in or out so it is not a complete invasion of their privacy but, that being said it is a frightening thought to think that it's a genuine option to increase productivity in business today.

Dangers and Implementations

The truth is that data will never be safe. Someone, somewhere, somehow will be able to hack into this data as no technology will be completely secure. Hackers are developing and evolving to become more intelligent and a study shown by CNN said that 47% of Americans have had their personal data exposed to a hacker in the year 2014 alone. The reality is, no data will ever be safe once it is put online.

My opinion

I do believe that it is unethical to use the private information of a user who may or may not be aware of what they are doing. For the purpose of advertising is understandable but this is far too easy to think that hackers can easily find out the same information. Once you post your personal data, thanks to GDPR, there are regulators that watch over what organisations can do with your personal data. I do not believe that it is possible for most people to go about their life by not indirectly sharing their personal data with the world and I don't see it getting any easier to protect it in the near future.

Conclusion

The primary aims of this report were to find out about how the data of a software developer is measured, how it is computed, what algorithms would be used and the ethics behind this data. The ways in which software engineers have managed to measure and assess its work has come on a long way since the 1960's. The computational and algorithmic methods continue to improve in making it easier to process data.

The major concerns are over data quality and the safety of data in relation to cyber-security. A dataset will ideally include all of the relevant data but we know this is not always the case but we try to keep it as clean as possible. By doing so it can give us a much more accurate result. In truth the are a lot more questions that could be asked in all of these cases as there more often than not, more than just one correct solution.

Bibliography

- https://books.google.ie/books?hl=en&lr=&id=lx_OBQAAQBAJ&oi=fnd&pg=PP1&dq=data+security+in+software+metrics&ots=_UhSRkXL-z&sig=lJt6p5jKiRsOabzwSfrAZ0i8RB4&redir_esc=y#v=onepage&q=data%20security%20in%20software%20metrics&f=false
- https://www.compact.nl/en/articles/data-quality-assessment/
- https://spacetimeinsight.com/five-aspects-of-data-quality/
- https://medium.com/@yupyork/the-best-developer-performance-metrics-6295ea8d87c0
- https://stackify.com/track-software-metrics/
- https://www.qasymphony.com/blog/64-test-metrics/
- https://hackernoon.com/all-you-need-to-know-about-gdpr-explained-8e336a1987ea
- https://money.cnn.com/2014/05/28/technology/security/hack-data-breach/index.html
- http://time.com/4062519/linkedn-spam-settlement/
- https://www.datasciencecentral.com/profiles/blogs/machine-learning-explained-understanding-supervised-unsupervised
- https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/
- https://www.quora.com/What-are-the-advantages-and-disadvantages-of-a-supervised-learning-machine
- http://www-public.imtbs-tsp.eu/~gibson/Teaching/CSC5524/CSC5524-PSP.pdf
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.138.6806&rep=rep1&type=pdf
- https://www.netguru.co/blog/comparison-automated-code-review-tools-codebeat-codacy-codeclimate-scrutinizer
- https://codeclimate.com/about/