Measuring Engineering

***To deliver a report that considers the ways in which the software engineering process can be measured and assesed in terms of measurable data, an overview of the computationial platforms available to perform this work, the algorithmic approachs available, and the ethics concerns surrounding this kind of analytics.***

In a competitive market, businesses and enterprises are constantly trying to bring the best product or service to the consumer at the best price, they are doing this as these businesses are competing in the market place to win over the consumer to their product, it is therefore also in the best interest of the consumer to get the best price for the product that they seek or price to quality ratio for that product if quality is a factor they are considering. Thus, stakeholders in enterprise are constantly seeking ways to automate and optimize elements of their production line to reduce the cost of production of their goods and services to allow themselves to compete effectively. It is this constant optimisation by the capitalist that incentivises skills and ultimately brings commodities to consumers for their best possible price.

For this reason, stakeholders in technology companies that produce software goods and services to consumers and other industries need to utilise some methods of attempting to optimise their production line. For a strictly software company, their production line is largely made up of the labour of those writing the code and programs for them. Apart from ensuring their staff are granted access to modern computing devices that inevitably reduce start up times, processing times, are overall more reliable and effective and providing the means to which the staff can all collaborate, the company also needs to ensure and enforce to some degree the efficiency of their individual staff. In the same way that the stakeholders of a stone masonry company would be interested in how many bricks a mason can clean and lay per hour/day for a given task or what progress this person is making to achieving the desired productivity, a software company is also interested in the quantity of work that their staff can produce per day. In this report, I will be detailing the ways in which the software engineering process can be measured and assessed in terms of measurable data, I will be giving an overview of the computational platforms available for this work, detailing some of the algorithmis approaches available for these measurements and finally giving my own view on the ethical concerns and responsibilities a software company should have when conducting these measurements.

While the assessment of productivity of the given example of a stone mason may be quite simple (most likely too simple) and have little variance on a weekly basis, given that they know how to correctly clean and lay the stone, how does one measure the productivity of a software developer? Does one simply assess that they can succesfully write software and simply count the lines of code they write and hope that that remains constant? Does this then leave possibility for disparity in different individuals software writing process not to be represented correctly in our simple analysis? I will now begin to detail the structure of the work performed in a software developing company and how that leads to available data that can be and is analysed by these industries of their staff to determine their staff’s productivity.

The development of software within a company begins with selecting the software process required for the service. In recent times, the “Agile” software process has seen increasing adoption to now be the more widely used process compared to the traditional “waterfall” process. ( source: <https://techbeacon.com/survey-agile-new-norm> ) The “Agile” process places emphasis on the continuous development of the software to meet changing needs and to cater for inevitable unforseen irregularities in the software design process that the traditional “waterfall” method enforces. The Agile development process allows for developers to plan and develop incrementally during the creation of the software. For measurability, this allows versions of software to be released frequently, each adding new elements of functionality, bug fixes and stability. The benefits of the Agile method over the traditional method are reducing the cost of being able to cater for changing customer needs, making it easier to get feedback with new versions able to be tested as well as generally overall a more rapid deployment of the service. One of the reasons for the Agile method to be so well adopted recently is the availability of data gathering and data analysis tools that can now almost replace project managers who’s job largely consists of attempting to estimate and plan the time frame for the development process. With frequently changing consumer needs and with only a single stable product being released in a waterfall model developed product/service, the use of Agile can be used instead to measure the overall software process time by means of measuring the time taken for individual stories and sprints to be completed. This combined with an estimate of the number of sprints and stories required for the next version of the software allows for more accurate analytics on the time taken for each version, how much functionality is added for each version and for measuring the progress of the quality of the product given by customer feedback.

One reason that could have seen a large shift toward the agile development process rather than the well documented waterfall process is the availability of powerful version control software. Version control software is essential to the modern software implementation process, it allows for development teams to all have access to a code base providing functionality of reverting to older versions of the code base, merging the contributions of multiple team members and many other functions. This technology being used in many tech projects has facilitated the unnecessity of large amounts of documentation that the waterfall method implemented. With commit comments and build versions as well as analytics that can be abstracted from the use of the software in calculating sprint and story times and making well calculated estimates, the old method of making estimates at design time results in better informed software estimations and measurements.

The most commonly used version control software that I mentioned in my last paragraph is git. Git facilitates a directory of files that any amount of contributors can alter and contribute to, in a company this will be the staff. The software will collect details on how many lines a contributor commited to the code base, how many commits each user made, the comments made about each commit, when the commit was made as well as lots of other pieces of data about the contributors and their individual commits and thus, contribution to the project as a whole.

1What data ?

-taxonomy

-code / commits

-Lines of code

-technical debt

Longitudually

Communications – email – slack

Consistency

Performance

Measurable data

2Computing

Analytics

Analytics as a service

Codeclimate

Case studies

3Algorithms

Computational intelligence

Algorithm

Machine learning

False positives from machine learning, bad trends, irrelevant data etc.

4Ethical concerns, wrap it up.