Section 1 – How one can measure Software Engineering Activity

There are several differing ways to measure software engineering activity, but all rely on having some sort of metric. There is no single best metric or method, but I will discuss some examples of different metrics used to measure software engineering in the following paragraphs.

Some metrics are based on the source code itself. A very simple example of this would be the number of lines of code a software engineer writes, and one could get something like the average number of lines that an engineer writes, modifies, or deletes a day. There are multiple ways of counting the lines of code. One is to count every physical line, which would be easier to do, but could lead to problems such as including lines of comment. Another is to count each logical statement as a line of code. This metric however is generally not considered very reliable, as it says nothing about the code's quality, and focuses purely on quantity.

The complexity of the code an engineer writes can also be used as a metric. Code complexity could be calculated in many ways and relies on other metrics, such as the number of lines of code, the number of unique operators and operands, the total frequency of operators and operands, the average number of live variables, and so on.

Another related metric is Impact. This metric measures the impact code changes has on the overall project and as a result depends on the amount of code created/changed, as well as the number of functions and/or files dependant on that code.

Another metric that can be used, although specific to agile development, is Sprint Burndown. Sprint Burndown is represented on a line graph, with the x-axis representing the days of the sprint, and the y-axis representing 'Remaining Effort' which is the estimated number of hours it will take to complete the remaining work at a given time. There is generally an Ideal remaining effort line which can be used to compare to the actual remaining effort.

Section 2 – Platforms on which this data can be collected, and calculations performed

An example of one platform or in this case an API that can be used to collect this data is the GitHub API. This can be used to get data on any engineer that has an account on the platform, any repository on the platform, and from private repositories if you use authentication. This API would relate to gathering data for metrics such as the number of lines of code, code complexity, impact, number of commits, number of pull requests, and so on. There are libraries for accessing the API in several languages such as Python, Java, JavaScript, Haskell, PHP, etc. The one downside of this is that it can only be used to gather the data. Calculations would have to be performed either by the person gathering the data or a different program. Similarly, another library/program would need to be used to display the data.

Software used for keeping track of tasks in agile development can also be used for gathering data for measuring software engineering activity. Platforms like Monday and Jira often have built in tools for getting graphs for metrics such as Sprint Burndown.

Section 3 – Computations that can be done over this data to profile the performance of Software Engineers

Section 4 – The ethics, legal and moral issues surrounding the collection of this data