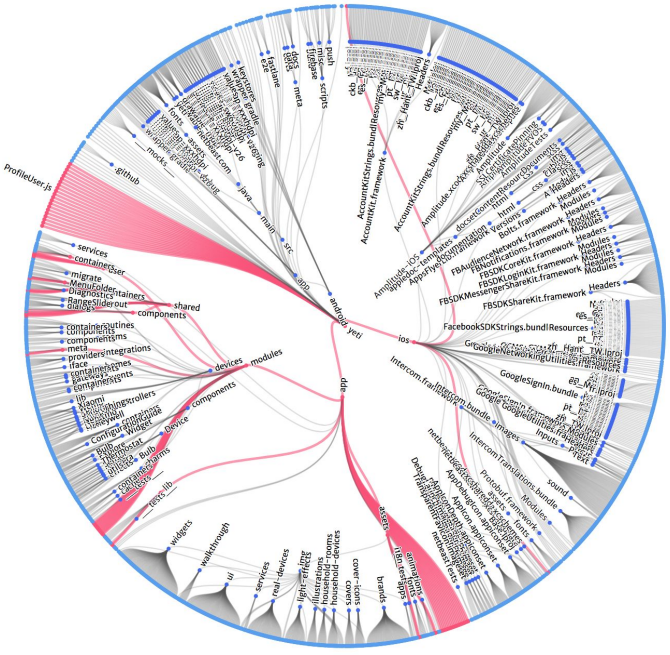
Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

**Module:** CSU33012
**Date:** 20th / Oct / 2019
**Name:** Mikhail Dyuldin

# Measuring Software Engineering

**Abstract:** To discover the ways in which a Software Engineer can be measured in terms of work ethic, average progress and other ways, followed by an overview of the computational platforms available for the performance of such work and the algorithmic approaches used. The report will then conclude with a discussion on the ethics surrounding this type of data gathering and interpreting.

Pictured Below: Visualization of a PR on Yeti Smart Home (https://getyeti.co) codebase[1]




[1] "Measuring Developer Productivity - By Jesus Dario." 9 Sep. 2019, https://hackernoon.com/measure-a-developers-impact-e2e18593ac79. Accessed 20 Oct. 2019.

# 1. Measurable Data

While there are many metrics that can be counted in the hopes that this will measure productivity, and this can be an overwhelming task to handle for an individual but the task becomes much more approachable when people consider these three metrics[2]:

### *1.2 - Value of Code*

This is a tricky metric to measure, especially since Software Engineers are taught to write code that can do more with less, meaning that they strive to accomplish the same task with the least amount of code written. Therefore simply counting the number of lines written won't cut it. A company called Static Object aims to tell you the value of your code by using a "Line Impact" algorithm to assess each line of code and give it a value, quantifying the impact that line of code has on the codebase and is able to track changes over time

This is much more complex than counting lines of code, but it also proves to be a more accurate measure of the value of code and gives a clearer picture in showing us the performance of a team or individual.

### *1.3 Speed*

Productivity must also be measured with regards to time constraints. It doesn't matter if a line of code has high impact or not if it took days to write, it would be like having a brilliant programmer that always missed deadlines.

Using Static Object as an example again, there included is a velocity measurement that allows a person to see a developers average amount of high impact lines of code per day.
These two metrics above don't give a complete view of a developer's progress. They must be weighed carefully in relation to how important the work being done currently is to the company, how close the deadline is and how much refactoring will be done in the future.

### *1.4 - Tech Debt*

Tech debt is probably the most elusive metric to measure, but first we must obtain an understanding of the term before we can continue and discuss the differences in tech debt and legacy code and how the two relate to each other. The concept of legacy code is that it is obsolete solutions / technologies that, for whatever reason, were preserved in the project, for example, PHP versions below 5.0. The spinning legacy gears in code is usually permeated within the nooks and crannies of projects. Changing something here can be scary, because the last person that knew how everything works, quit many years ago.

Any software system has a certain amount of essential complexity required to do its job, but most systems contain cruft[3] that makes it harder to understand. Technical debt is a metaphor, coined by Ward Cunningham in 1992, that frames how to

[2] "3 Key Metrics to Measure Developer Productivity - GitClear ...." 28 Sep. 2017, https://medium.com/static-object/3-key-metrics-to-measure-developer-productivity-c7cec44f0f67. Accessed 20 Oct. 2019.

[3] "TechnicalDebt - Martin Fowler." 21 May. 2019, https://martinfowler.com/bliki/TechnicalDebt.html. Accessed 21 Oct. 2019.

think about dealing with this cruft. The technical debt metaphor treats the cruft as a debt, whose interest payments are the extra effort these changes require.

A quick solution today might solve an immediate problem, but create other problems that need to be solved in the future. Examples of such problems are: Code duplication and complexity, lack of documentation, and programming rule violations. There is no single formula that can calculate tech debt but identifying the debt sources and to estimate the amount of added development time is a good place to start.

Referencing Static Object again, as it continues to add new features and Integrations, the ability to measure technical debt is beginning to look like a reality. The tool will identify where time is spent refactoring code after a project is considered “done”. The work done in this time is roughly equivalent to tech debt.

# 2. Commercial Platforms

## *2.1 - Is it Worth it?*

Would there be a reason to do all of this measuring if we assume that every programmer on average operates around the same efficiency? Anecdotally speaking, there have been many accounts by other Software Engineers where they regularly report seeing people that work 10x more efficiently in comparison to their peers. As John D.Cook, PhD and president of his consulting company, put it in a blog post in January of 2011.[4]

*“I’ve seen programmers who were easily 10x more productive than their peers. I imagine most people who have worked long enough can say the same. I find it odd to ask for academic support for something so obvious.* ***Yes, you’ve seen it in the real world, but has it been confirmed in an artificial, academic environment?****”*

Steve McConnell, Author of various Software Engineering textbooks and speaker, chose to carry out research into the 10x developer, to prove whether it is fact of fiction. The article in question is *“Origins of 10x - How valid is the underlying research?”.*[5]

He concludes *“The body of research that supports the 10x claim is as solid as any research that’s been done in Software Engineering”.*

So now that we know that these “Rockstar” programmers exist, we need to look into what companies are trying to analyse programmers to find the top 20%, and are these analytics providing any meaningful results? Since Software Engineers are not paid by performance, companies have a very high incentive to hire only the best.

## *2.2 - GitPrime*

GitPrime is a self proclaimed industry standard in Performance Management, boasting a large array of popular companies who use GitPrime to succeed in optimising their teams. Examples of

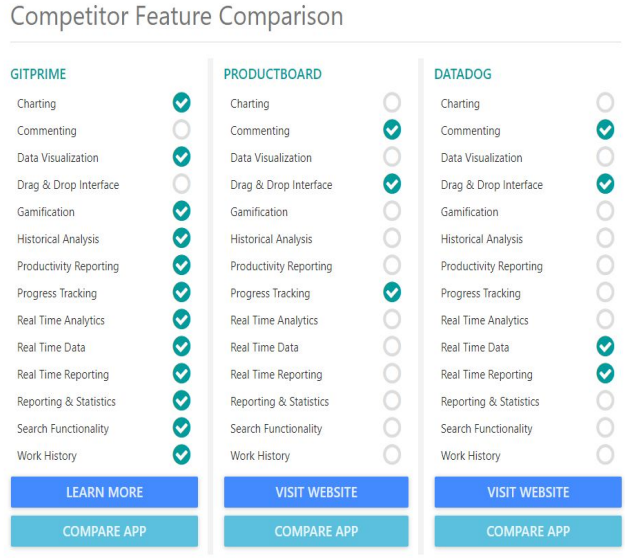
[4] "Some programmers really are 10x more productive." 10 Jan. 2011, https://www.johndcook.com/blog/2011/01/10/some-programmers-really-are-10x-more-productive/. Accessed 26 Oct. 2019.

[5] "The Origins of 10x - How Valid is the Underlying Research ...." https://www.construx.com/blog/the-origins-of-10x-how-valid-is-the-underlying-research/. Accessed 26 Oct. 2019.

such companies are : Adobe, PayPal, Disney, VMware, Atlassian, and many more.

GitPrime aims to provide easy to read visualisations to management in order for them to be able to make quick decisions based on how their goals are being made, how impactful their code is, real-time analytics and much more. Because this is a tool designed for management, reviews say that it is hard to convince developers to use this product, and the product itself carries a very significant pricing amount of 750$ per month, much more than a lot of startups or smaller companies can afford. This shows us just how much of a premium service production analytics is.

GitPrime can afford to be this expensive because it doesn't seem to have any other good quality competitors.[6]




If you only look at the amount of features other applications offer.

## *2.3 - Wildly Different Approaches*

There are of course many other aspects of an employee that you can measure[7]:

- Online presence
- Work hours
- Holidays
- Personal details
- Pay + Expenses claimed
- Interaction with training
- Personal Software Process (PSP)
- Programs used through the day
- Email/Slack usage
- Fitbit/Sensor++
- Office space temperature
- Happiness

Above are some of the metrics you can expect companies to attempt to track.

The company called Humanyze has brought in a badge that an employee can wear which records video, voice / tone of voice, conversations and location data, uploading around 4GB of data every day.

Steelcase attempts to improve employees health by monitoring their posture and other health metrics, such as heart rate, breathing and stress levels through a specialised office chair. All of this information is then displayed on an iPad, notifying the employee to take standing breaks or to correct their posture, potentially forcing the employee to do breathing exercises and stretches.

Personally, I have a bone to pick with the above two companies and their methods. I will discuss the ethics of such practice in the Ethics section of this report, and I will show how this might negatively impact workers in ways they might not have

[6] "GitPrime Features & Capabilities | GetApp®." https://www.getapp.com/business-intelligence-analytics-software/a/gitprime/features/. Accessed 29 Oct. 2019.

[7] "Stephen Barrett - School of Computer Science and Statistics." https://www.scss.tcd.ie/Stephen.Barrett/. Accessed 6 Nov. 2019.

anticipated, and how in the end the negatives will probably outweigh the positives.

# 3. Algorithmic Approach

## *3.1 - Halstead's Metrics*

The problem of evaluating the quality of software products has been the subject of many studies, however, their object, as a rule, is the Professional Software Projects (PSP). The problem of assessing the quality of the Academic Software Projects (ASP) is given much less attention, although this task inevitably arises in technical specialties, where students create a large number of programs during the educational process. Due to the nature of the ASP, special tools are required to process such large arrays of project documentation.

Halstead's Software Engineering metrics are used in many commercial and academic applications today that count software lines of code[8].

By counting the tokens and determining the amount of distinct operators and operands and the total occurrences of them throughout the code.

n1 = Number of distinct operators.
n2 = Number of distinct operands.
N1 = Total number of occurrences of operators.
N2 = Total number of occurrences of operands.

In addition to the above, Halstead defines the following :

n1* = Number of potential operators.
n2* = Number of potential operands.

"Halstead refers to n1* and n2* as the minimum possible number of operators and operands for a module and a program respectively. This minimum number would be embodied in the programming language itself, in which the required operation would already exist (for example, in C language, any program must contain at least the definition of the function main()), possibly as a function or as a procedure: n1* = 2, since at least 2 operators must appear for any function or procedure : 1 for the name of the function and 1 to serve as an assignment or grouping symbol, and n2* represents the number of parameters, without repetition, which would need to be passed on to the function or the procedure." -GeeksforGeeks.org

Halstead's Algorithmic approach as described on GeeksforGeeks [9]:

**Halstead Program Length** – The total number of operator occurrences and the total number of operand occurrences.
N = N1 + N2
And estimated program length is,
n1log2n1 + n2log2n2

The following alternate expressions have been published to estimate program length:

- NJ = log2(n1!) + log2(n2!)

[8] "Software Engineering | Halstead's Software Metrics ...." https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/. Accessed 6 Nov. 2019.

[9] "Software Engineering | Halstead's Software Metrics ...." https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/. Accessed 6 Nov. 2019.

- NB = n1 * log2n2 + n2 * log2n1
- NC = n1 * sqrt(n1) + n2 * sqrt(n2)
- NS = (n * log2n) / 2

**Halstead Vocabulary** – The total number of unique operator and unique operand occurrences.
n = n1 + n2

**Program Volume** – Proportional to program size, represents the size, in bits, of space necessary for storing the program. This parameter is dependent on specific algorithm implementation. The properties V, N, and the number of lines in the code are shown to be linearly connected and equally valid for measuring relative program size.
V = Size * (log2 vocabulary) = N * log2(n)

The unit of measurement of volume is the common unit for size “bits”. It is the actual size of a program if a uniform binary encoding for the vocabulary is used. And error = Volume / 3000

**Potential Minimum Volume** – The potential minimum volume V* is defined as the volume of the most succinct program in which a problem can be coded.
V* = (2 + n2*) * log2(2 + n2*)

Here, n2* is the count of unique input and output parameters

**Program Level** – To rank the programming languages, the level of abstraction provided by the programming language, Program Level (L) is considered. The higher the level of a language, the less effort it takes to develop a program using that language.
L = V* / V

The value of L ranges between zero and one, with L=1 representing a program written at the highest possible level (i.e., with minimum size).
And estimated program level is L^ =2 * (n2) / (n1)(N2)

**Program Difficulty** – This parameter shows how difficult to handle the program is.
D = (n1 / 2) * (N2 / n2)
D = 1 / L
As the volume of the implementation of a program increases, the program level decreases and the difficulty increases. Thus, programming practices such as redundant usage of operands, or the failure to use higher-level control constructs will tend to increase the volume as well as the difficulty.

**Programming Effort** – Measures the amount of mental activity needed to translate the existing algorithm into implementation in the specified program language.
E = V / L = D * V = Difficulty * Volume

**Language Level** – Shows the algorithm implementation program language level. The same algorithm demands additional effort if it is written in a low-level program language. For example, it is easier to program in Pascal than in Assembler.
L’ = V / D / D
lambda = L * V* = L2 * V

**Intelligence Content** – Determines the amount of intelligence presented (stated) in the program This parameter provides a measurement of program complexity, independently of the program language in which it was implemented.
I = V / D

**Programming Time** – Shows time (in minutes) needed to translate the existing algorithm into implementation in the specified program language.
T = E / (f * S)
The concept of the processing rate of the human brain, developed by the psychologist John Stroud, is also used. Stoud defined a moment as the time required by the human brain requires to carry out the most elementary decision. The Stoud number S is therefore Stoud's moments per second with:
5 <= S <= 20. Halstead uses 18. The value of S has been empirically developed from psychological reasoning, and its recommended value for programming applications is 18.

Stroud number S = 18 moments / second

seconds-to-minutes factor f = 60

### *3.2 - Example*

Here is an example implementation of Halstead's methodology as seen on geeksforgeeks.org[10]

**Example** – List out the operators and operands and also calculate the values of software science measures like

```
int sort (int x[ ], int n)

{
    int i, j, save, im1;
    /*This function sorts array
x in ascending order */
    If (n< 2) return 1;
    for (i=2; i< =n; i++)
    {
        im1=i-1;
        for (j=1; j< =im1; j++)
            if (x[i] < x[j])
            {
                Save = x[i];
                x[i] = x[j];
                x[j] = save;
            }
    }
    return 0;
}
```

| Operato-rs | Occurre-nces | Operan-ds | Occurre-nces |
|---|---|---|---|
| int | 4 | sort | 1 |
| () | 5 | x | 7 |
| , | 4 | n | 3 |
| [] | 7 | i | 8 |
| if | 2 | j | 7 |
| < | 2 | save | 3 |
| ; | 11 | im1 | 3 |
| for | 2 | 2 | 2 |
| = | 6 | 1 | 3 |
| - | 1 | 0 | 1 |
| <= | 2 | - | - |
| ++ | 2 | - | - |

[10] "Software Engineering | Halstead's Software Metrics ...." https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/. Accessed 6 Nov. 2019.

| return | 2 | - | - |
|---|---|---|---|
| {} | 3 | - | - |
| n1=14 | N1=53 | n2=10 | N2=38 |

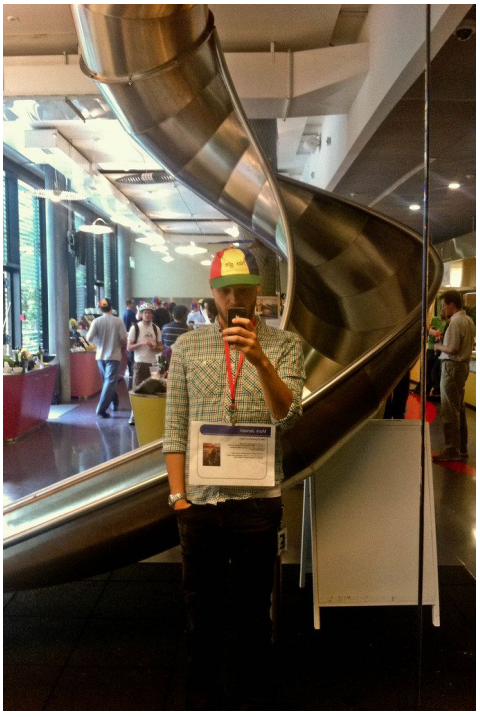
Therefore,
N = 91
n = 24
V = 417.23 bits
N^ = 86.51
n2* = 3 (x:array holding integer to be sorted. This is used both as input and output)
V* = 11.6
L = 0.027
D = 37.03
L^ = 0.038
T = 610 seconds

# 4. Ethics

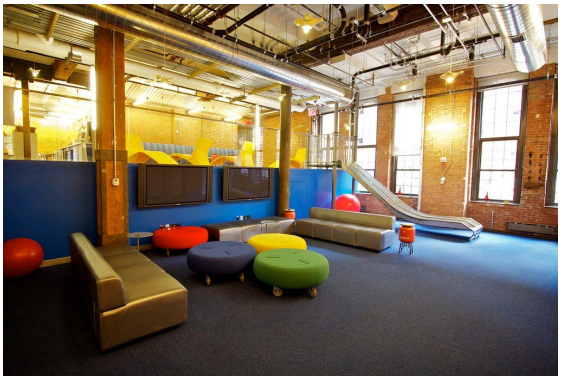
### *4.1 - GitPrime Ethics*

GitPrime has shown it's investors and consumers that productivity increases a very large amount, once their product is properly utilised, mostly attributed to the managers being able to manage the projects much more effectively with the new tools provided. As such I do not think GitPrime's approach violates any ethics.

In order for productivity to increase so much, it requires the managers to be up to date on the work being done and to successfully divert the work as needed, and in order for productivity to maintain a high level, it is well known by many in the industry that the happiness of your Software Engineers plays a very big part in how productive they are. Companies know this and it is a big reason behind why a lot of IT offices look like a mix between comfort and a children's playground.



Pictured - A slide connecting floors in one of Google's offices.



Pictured - A playground-like lounge area in one of Google's offices

As you can see from the above pictures, Google and other companies prioritise heavily into the comfort of their workers. This is mostly due to the benefits it has on work and helps reduce turnover rates of a company, while also making them look progressive.

### *4.2 - Humanyze and Steelcase*

Humanyze and Steelcase have taken micro-management to the next level. If daily hour long meetings, scrutinising you work and changing direction twice a day wasn't enough, then these will. With a badge that reminds you of something straight out of George Orwell's book, 1984, and a chair that feels your every slouch and every raised heart rate, sending the data to your superiors for them to judge you on every habit that you may have while they, presumably, sit on a regular, non-authoritarian office chair.

If that doesn't sound bleak enough for you to want to move into a cabin in the woods then I don't know what will.

"Those concerned about their privacy might be alarmed by the arrival of such badges. But Humanyze says it doesn't record the content of what people say, just how they say it. And the boss doesn't get to look at individuals' personal data. It is also up to the employee to decide whether they want to participate."[11] - Said the Washingtonpost.com, While in the same article the Chief Executive of Humanyze said the following, ""Within three or four years, every single ID badge is going to have these sensors," predicted Ben Waber, chief executive of Humanyze, a Boston-based employee analytics company. "We are only scratching the surface right now.""
The whole article is full of mixed messages, some of their researchers saying that giving a choice to the employee on whether they want to participate in the work surveillance or not is essential in order for the benefits to outweigh the negatives, and then there are other messages saying that this will be standard in every workplace.

My take from the above is that they are trying to ease this onto people so that the majority of the workforce is using the product "voluntarily" and then after a couple of years they can pressure the rest into using them as a mandatory requirement.

Facebook also said that they didn't look at, sell or do anything purposefully nefarious with user's data, but now, almost on a daily basis we hear more emails between Google, Facebook and other big conglomerates, discussing how to use this data to their benefits and using them as bargaining chips in their deals. Emails as far back as 2012[12] have surfaced, implying that this isn't a practice that will stop anytime soon because it's such a common practice for them at this point.

Capitalism has shown us time and time again that companies only hold the values of their shareholders in mind, and we cannot trust them to keep their word when it comes to privacy. The only way we can be sure that our privacy is not being infringed is by having the software completely open source, available to the scrutinizing eye of the public, because the public is the only entity that can be trusted in keeping the public's best interests in mind.

---

[11] "This employee ID badge monitors and listens to you at work — except ...." 7 Sep. 2016, https://www.washingtonpost.com/news/business/wp/2016/09/07/this-employee-badge-knows-not-only-where-you-are-but-whether-you-are-talking-to-your-co-workers/. Accessed 7 Nov. 2019.

[12] "Facebook discussed using people's data as a bargaining chip ...." 30 Nov. 2018, https://www.washingtonpost.com/technology/2018/11/30/facebook-used-peoples-data-bargaining-chip-emails-court-filings-suggest/. Accessed 7 Nov. 2019.

# 5. Bibliography


1. "Measuring Developer Productivity - By Jesus Dario." 9 Sep. 2019, https://hackernoon.com/measure-a-developers-impact-e2e18593ac79. Accessed 20 Oct. 2019.
2. "3 Key Metrics to Measure Developer Productivity - GitClear ...." 28 Sep. 2017, https://medium.com/static-object/3-key-metrics-to-measure-developer-productivity-c7cec44f0f67. Accessed 20 Oct. 2019.
3. "TechnicalDebt - Martin Fowler." 21 May. 2019, https://martinfowler.com/bliki/TechnicalDebt.html. Accessed 21 Oct. 2019.
4. "Some programmers really are 10x more productive." 10 Jan. 2011, https://www.johndcook.com/blog/2011/01/10/some-programmers-really-are-10x-more-productive/. Accessed 26 Oct. 2019.
5. "The Origins of 10x - How Valid is the Underlying Research ...." https://www.construx.com/blog/the-origins-of-10x-how-valid-is-the-underlying-research/. Accessed 26 Oct. 2019.
6. "GitPrime Features & Capabilities | GetApp®." https://www.getapp.com/business-intelligence-analytics-software/a/gitprime/features/. Accessed 29 Oct. 2019.
7. "Stephen Barrett - School of Computer Science and Statistics." https://www.scss.tcd.ie/Stephen.Barrett/. Accessed 6 Nov. 2019.
8. "Software Engineering | Halstead's Software Metrics ...." https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/. Accessed 6 Nov. 2019.
9. "Software Engineering | Halstead's Software Metrics ...." https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/. Accessed 6 Nov. 2019.
10. "Software Engineering | Halstead's Software Metrics ...." https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/. Accessed 6 Nov. 2019.
11. "This employee ID badge monitors and listens to you at work — except ...." 7 Sep. 2016, https://www.washingtonpost.com/news/business/wp/2016/09/07/this-employee-badge-knows-not-only-where-you-are-but-whether-you-are-talking-to-your-co-workers/. Accessed 7 Nov. 2019.
12. "Facebook discussed using people's data as a bargaining chip ...." 30 Nov. 2018, https://www.washingtonpost.com/technology/2018/11/30/facebook-used-peoples-data-bargaining-chip-emails-court-filings-suggest/. Accessed 7 Nov. 2019.
13.