Measuring Software Engineering

**Introduction**

**Measurable data**

* **Number of bugs per 1000 lines of code**

The industry average of bugs per 1,000 line of code is between 15 and 50 bugs, this figure depends though on when the code is actually typed. The basic human error rate is on average 5%. This figure does depend on many different factors such as the level of testing within the code.

* **Level of testing**
* **Lines of code**
* **Test Coverage / code coverage**
* **Leadership / mentorship**
* **Customer satisfaction**
* **Line numbers why and why not to use it**

**Computational platforms available**

* **Github**
* **Version Control**
* **Commits and version control**
* **Slack / Teams**
* **Trello**

**Algorithmic approach**

* **Cycle time**

Cycle time is defined as “The total time that elapses from the moment when the work is started on a task until its competition”. The difference between lead time and cycle time is that cycle time is from when the tasked is created through till the work is completed whereas the cycle time is from when the work is started till the time the work is completed. The cycle time data can be used for future projects for gauging how long it’s going to take a certain team to complete a required task. It can also be used to gauge progress in seeing progression of seeing if your cycle time is getting shorted. If your cycle time is shortening over time it’s a sign of good efficiency. Measuring your cycle time is a very straight forward process. All you need to do us record when the task was started and when the task was completed.

* **Waterfall**

It is a fairly straight forward to implement in a new software project. It is a step by step process. There is six stages to waterfall methodology. They are :

1. Requirements

During this initial phase, the requirements of the project are noted and taken down. The document that it gets written on is called a requirements document. It defines what the application should do but not how it should do it.

1. Analysis

During this stage the system gets analysed in order to properly generate the models and business logic that will get used in the application.

1. Design

This stage covers the technical design requirements. In this stage you choose which programming language to use and technologies that you will use in the process. At the end of this stage a design specification report will be generated in which it explains how exactly the logic will be technically implemented.

1. Coding

In this stage the source code is written. This will implement all models and logic that were specified in previous stages

1. Testing

This stage involves QA’s(Quality Assurance) and beta testers to test the source code that’s been written to try and find any bugs within the code. These people report back with any bugs that are found.

1. Operations

In the operations stage the application that has been developed is now ready to be released into a live environment. This stage also involves support and maintenance in order to keep the application up to date and free from bugs.

* **Agile methods**

Agile methods are used to find ways to enhance the software development process, both lead times and cycle times are taken into account. The lead time is the time taken by a team to generate ideas, then develop that idea before delivering the software product, whereas the cycle time is the time from developing the software and deploying it into production.

* **Scrum**

Scrum is a framework within which people can address complex adaptive problems while creating and delivering products to a customer at the highest possible value. Scrum involves a team working together. A scrum is the actual time period when the team works together to finish an increment. The team completes a scrum cycle which can range anywhere from one week to a four week cycle. Within that cycle the team will start off with a sprint planning in which they will set our their targets and the time they want to have it completed by. These goals are set out by the scrum master. By the end of the meeting the full team should be clear on what can be delivered in the sprint and how the increment can be delivered. The team usually will have a daily scrum which will involve a short meeting of 15 minutes in which they reflect on the work that’s been produced in that day. This helps ensure that each member of the team is on the same page on what needs to be done. The team then has a sprint review at the end of the sprint. In this review they review the work that’s been carried out. The last part of the scrum is called sprint retrospective. The goal is that after the sprint cycle the team will either increment(show what’s been done in the scrum) of demo the project they have completed to the client. The team will also talk about what worked and didn’t work in the scrum. This creates a place where the team can focus on what needs to be improved for the next time.

* **Algorithm Analysis**

**Ethics concerns**

**Bibliography**

<https://diceus.com/top-7-software-quality-metrics-matter/>

<https://www.quora.com/What-is-the-average-ratio-of-bugs-to-a-line-of-code>

<https://www.scrum.org/resources/what-is-scrum>

<https://www.atlassian.com/agile/scrum>

<https://www.cgi.com/us/en-us/life-sciences/blog/12-principles-of-agile-methodologies>

<https://airbrake.io/blog/sdlc/waterfall-model>

<https://screenful.com/blog/software-development-metrics-cycle-time>