Software Engineering process: Measurement and Assesment

# Objective

*‘The objective of this assignment is to deliver a report that considers the ways in which the software engineering process can be measured and assessed in terms of measurable data, an overview of the computational platforms available to perform this work, the algorithmic approaches available and the ethical concerns surrounding this kind of analytics.’*

In this report I will aim to examine and explore academic material on the subject of software engineering as well as refer to the teachings of Professor Stephen Barrett who conducts the module which I study, entitled CS3012:Software Engineering.

# Introduction

In order to fully analyse ways in which the software engineering process can be measured and assessed we must first look at what software engineering is. Software engineering by definition is ‘a detailed study of engineering to the design, development and maintenance of software.’ (Times, n.d.) It is a process which takes the end users need into account by applying engineering principles in order to develop software. It uses a variety of programming languages in order to design, construct and test end user applications. (Techopedia, n.d.) The role of a software engineer is to not only create and develop new software but also take and modify existing software to suit their needs. A software engineer will often first think of the solution and later think of the technology needed to implement it. They can be seen as builders and designers who many a time encounter issues with their programs that they never anticipated, yet they use logical reasoning and exploration to overcome them. Often times, a software engineer’s primary role is debug and fix code. As software engineer Eric Elliott once said ‘Software developers write and fix bugs for a living. Sometimes some software gets made, too.’ (Elliott, 2017)

The term software engineering was first introduced in the 1968 at a conference organized by NATO to manage the software crisis. This refers to the difficulties which were encountered in developing large and complex systems throughout the 1960’s. The main reasoning behind this development was that costs could be reduced as well as being able to create more reliable software. (Sommerville, 2008) Although a young discipline, software engineering has undergone rapid developments and is constantly evolving.

Today, software engineering plays a pivotal role in society. It impacts all our lives on a daily basis through the various technologies we engage with. It is an exciting young, discipline that is responsible for so much of our world today. In order to truly understand the process, I will break down this report and analyse the following.

1. The ways in which software engineering can be measured and assessed in terms of measurable data.
2. The computational platforms available to perform this work.
3. The algorithmic approaches available to deal with this data.
4. The ethics surrounding this topic.

# Software engineering process

In order to be better able to analyse and discuss the other topics in this report it is fundamental that the software engineering process is understood. The software engineering process also known as the software engineering methodology is the method used to process, plan and structure and structure the development of information system. (wikibooks, n.d.)Developed in the 1960’s around the same time as the introduction of software engineering it is one of the oldest and formal methodology for building an information system. It is the model chosen for designing and overseeing the creation of new software from the beginning to the end of the production. An important thing to note about the software engineering process is that there is a no one size fits all solution with many different methodologies available to suit a wide variety of projects.

One well known process is the systems development life cycle (SDLC) which is composed of a number of steps and clear working phases. SDLC provides a detailed plan which allows to specific software to be developed, altered and enhanced. This can be broken down into six main steps which are demonstrated in the image below. (BusinessField, n.d.)



There are a number of different types of SDLC model which are all based on the same framework. Different methods suit different types of projects. The most common methods are as follows:

* Waterfalls model: This model is based on a linear sequential downwards flow.
* V-shaped model: This is an extension of the waterfalls model, however instead of moving linearly downwards the process is bent upwards following the implementation phase (stage 4).
* Prototyping model: This model is designed around creating prototypes of software applications.
* Spiral Model: This method combines the advantages of top down and bottom up concepts as well as elements of both the waterfall and prototyping model.
* Iterative and incremental model: Similar to the waterfall model however it finishes with the deployment of cyclical interactions.
* Agile model: This model is based on the iterative and incremental model and evolves through collaboration between a cross functional team.

There are several other models which are not mentioned here, and these are used to deal with a variety of tasks or activities when required. When such processes are not evoked, or software projects are not well managed software engineering projects can easily run over time or exceed their budget. This is why there is such a huge importance placed on software engineering processes in today’s business orientated, fast paced world.

# Measurable Data

The first issue I wish to tackle in this report is that of measurable data. Measurable data is