Assignment 1

Lorenzo Tecchia

5581906

2023.04.24

Contents

1 Organizational 3

2 Conflicting goals in software development 4

3 Clarification of terms 6

Chapter 1

Organizational

I've successfully done all the point of this introductory task.

Chapter 2

Conflicting goals in software development

(a) All four goals cannot be achieved equally at their maximum capabilities, rather it should be a balanced distribution of them. If quality was taken as example; it would be impossible to reach a great level of quality whilst keeping the costs low and the elapsed time to completion very short. Also having a great scope for the software and achieving great quality would in any case take a lot of time to be fully complete.[6]

(b) A piece of software that served the purpose of providing programming classes and exercises for third-world countries would have priority in keeping low the maintenance costs and the running costs.
A piece of software with the intention of computing molecules for the covid-19 vaccine has like main goal to be developed in the shortest time possible in order decrease the level of mortality of the virus.
A software with the goal of evaluating x-ray scans for potential cancer masses, has to be of the greatest quality possible to provide a near to perfect diagnosis of the patient.
An University Management software has to provide multiple functions: a class management system, a class scheduling system, a calendar system or also an archive and a catalogue of many courses and exams. So its scope should be large enough to incorporate all these requirements.

(c) It's possible to combine the most important goals for software development with a constraint triangle as seen in the figure 2.1[1]. At the vertexes there are: Cost,Time and Scope with the central goal being the quality of the software. The principle behind this triangle is that vertexes of the triangle must always be developed equally together. Scope is directly proportional to Time and Cost, since increasing the scope necessary means increasing the cost and the time of the project. Time and Cost, on the other hand, are inversely proportional

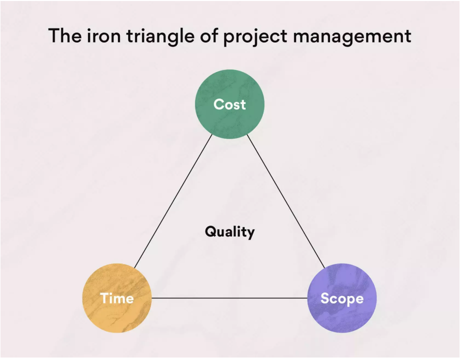

Figure 2.1: Triangle Constraint

since, cutting one down means increasing the other and vice versa. The tight correlation between these 3 key aspects makes it so that altering one goal cause a corresponding change in the others.

(d) As discussed earlier Time and Cost in a project are inversely proportional. Cutting budget on production and human resources means more time spent developing the amount of work that could have been done by many more people working on the same project.
Assuming that a software developer would need 1 year development time for a software with defined scope and quality, one would like to understand how long it takes 2/5/150 developers to execute the same project. Making a note about the relation between cost and time-it's safe to say that the scenario in which there is only one developer working on the project, it's the one with the least cost. Following the principles stated above, the scenario with 150 developers is the one with the highest cost.
Continuing with the subject of # of developer vs time spent developing; often times increasing the amount of people working on the same project could come with some trade offs. More people working means also more potential mistakes and more time spent coordinating the various parts working together; thus plateauing the benefits from the increased workforce. More people working on the same project also means to define various standards to be able to understand fluently each others work. This can also very frequently happen with even just 2 people working on the same project. [4]

Chapter 3

Clarification of terms

(a) Class: It's a blueprint of Object with the same name, with the inclusion of its attributes and methods.[3]

```
public class Article {
    // some code
     }
```

Inheritance: The passage of same attributes and methods between the parent class and the child class

```
public class Magazine extends Article{
        // some code
        }
```

Attribute: The quality components of a class.

```
public class Article {
    private String doi_A;
    private String title;
    private String accessMode;
    private String editor;
    private String topic;
}
```

Operation:If we are referring to the term in the context of O-O programming: An action done to/from an object. The act of executing a method.

```
 // some code
 article.getName();
 //some code
```

More in general, an operation is the action of functioning or the fact of being active or in effect. (e.g. the company's first hotel is now in operation)[2]

Method: A function inside the scope of a Class that defines the action that a Class can do.

```java
   public class Article {
   public String getTitle() {
       return title;
   }
}
```

Copy: Two instances of an object with same name, same attributes, same methods.

```java
 class Article {
  private String article;

  public Article(Article article2) {
    this.article = article2.article;
  }
}
```

Object: An instance of a Class.

```java
ArrayList<Article> articles = new ArrayList<>();
```

Library: A collection of files with different classes.

```java
import java.util.ArrayList;
```

Specification: A request made from the customer of a software project.(e.g. A GUI based Software needs to implement a GUI and the presence of a GUI in said software is a specification).[5]

Implementation: If we are referring to the term in the context of O-O programming: The actual piece of code of an Interface.

```java
public class ImplementazioneArticle implements ArticleDAO {
   // some code
}
```

More in general and implementation is the execution/application of a plan.[2]
Verification: Is the process of checking that a type of Software meets its specifications. (e.g. The process of checking that the software has a GUI, if it's requested by its specification).[5]

(b) A Class is the the ”blueprint” of an object. It defines which attribute can have and which action (methods) it can perform. It encompass every type of object with the same specifications. Objects are instances of classes, so they have specified attributes -even though they have the same blueprint-they can differ between one an other.

```java
public class Article{ // class Article
    private String name; // attributes
    private String date;

    public Article(String name, String date){ // constructor of the class
        this.name = name;
        this.date = date;
    }
}

// some code

// two instances of the class Article, therefore objects
Article article1 = new Article("name1", "date1"); // object 1
Article article2 = new Article("name2", "date2"); // object 2
```

(c) A part from all the strictly O-O Programming terms like: Class, Attribute, Method, ecc... The only two terms that are not covered would be Operation and Implementation. Which are the act of maintaining a project in execution and the application of a plan, respectively.

Bibliography

Asana, Team. *Project Managment Triangle.*

Collins English dictionary & thesaurus. 4th ed. Glasgow: HarperCollins, 2011. ISBN: 9780007337460 (pbk.)

Deitel, Harvey M. and Paul J Deitel. *Java: how to program.* 6th ed. Upper Saddle River, NJ: Deitel, 2005. ISBN: 0131483986.

Ghezzi, Carlo. *Fundamentals of software engineering / Carlo Ghezzi ; Mehdi Jazayeri ; Dino Mandrioli.* eng. Englewood Cliffs, NJ: Prentice Hall [u.a.], 1991. ISBN: 0-13-820432-2.

"IEEE Standard for System, Software, and Hardware Verification and Validation". In: *IEEE Std 1012-2016 (Revision of IEEE Std 1012-2012/ Incorporates IEEE Std 1012-2016/Cor1-2017)* (2017), pp. 1–260. DOI: `10.1109/IEEESTD.2017.8055462`.

Sommerville, Ian. *Software engineering environments.* Vol. 7. London, U.K.: P. Peregrinus, on behalf of the Institution of Electrical Engineers, 1986. ISBN: 0863410774.