**Experiment No.02**

PART A

(PART A: TO BE REFFERED BY STUDENTS)

**A.1 Aim:** Categorize projects using COCOMO, and estimate effort and development time required for a project

**A.2 Prerequisite:** Software Development Life Cycle /Literature survey

**A.3 Outcome:**

**After successful completion of this experiment students will be able to know**

1. How to estimate **cost** for a software project
2. How to estimate **effort** for a software project
3. How to estimate **duration** for a software project

**A.4 Theory:**

**Project Estimation Techniques**

A software project is not just about writing a few hundred lines of source code to achieve a particular objective. The scope of a software project is comparatively quite large, and such a project could take several years to complete. However, the phrase "quite large" could only give some (possibly vague) qualitative information. As in any other science and engineering discipline, one would be interested to measure how complex a project is.

One of the major activities of the project planning phase, therefore, is to estimate various project parameters in order to take proper decisions. Some important project parameters that are estimated include:

**Project size:** What would be the size of the code written say, in number of lines, files, modules?

**Cost:** How much would it cost to develop a software? A software may be just pieces of code, but one has to pay to the managers, developers, and other project personnel.

**Duration:** How long would it be before the software is delivered to the clients?

**Effort:** How much effort from the team members would be required to create the software?

**COCOMO**

COCOMO (Constructive Cost Model) was proposed by Boehm. According to him, there could be three categories of software projects: organic, semidetached, and embedded. The classification is done considering the characteristics of the software, the development team and environment. These product classes typically correspond to application, utility and system programs, respectively. Data processing programs could be considered as application programs. Compilers, linkers, are examples of utility programs. Operating systems, real-time system programs are examples of system programs. One could easily apprehend that it would take much more time and effort to develop an OS than an attendance management system.

The concept of organic, semidetached, and embedded systems are described below.

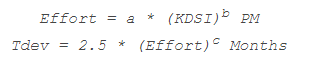
Organic: A development project is said to be of organic type, if the project deals with developing a well understood application. The development team is small. The team members have prior experience in working with similar types of projects

Semidetached: A development project can be categorized as semidetached type, if the team consists of some experienced as well as inexperienced staff. Team members may have some experience on the type of system to be developed

Embedded: Embedded type of development project are those, which aims to develop a software strongly related to machine hardware. Team size is usually large.

Boehm suggested that estimation of project parameters should be done through three stages: Basic COCOMO, Intermediate COCOMO, and Complete COCOMO.

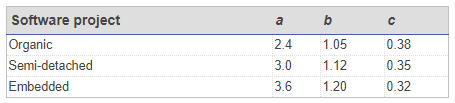
**Basic COCOMO Model**

The basic COCOMO model helps to obtain a rough estimate of the project parameters. It estimates effort and time required for development in the following way:   


Where,

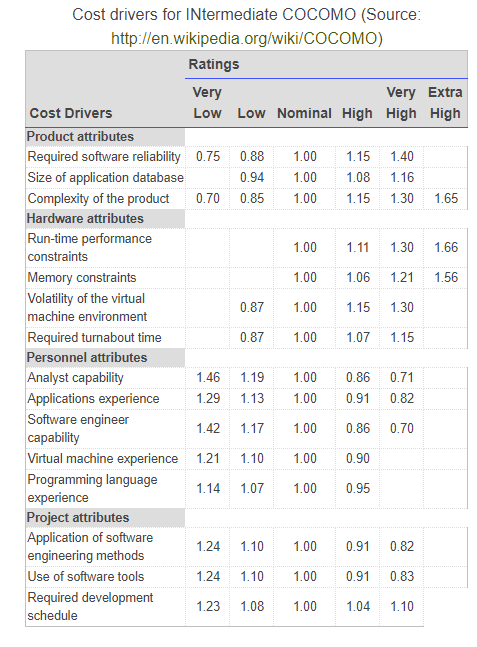
* KDSI is the estimated size of the software expressed in Kilo Delivered Source Instructions.
* a, b, c are constants determined by the category of software project
* Effort denotes the total effort required for the software development, expressed in person months (PMs)
* Tdev denotes the estimated time required to develop the software (expressed in months)

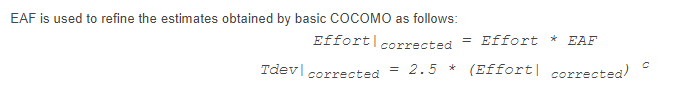
The value of the constants a, b, c are given below:



**Intermediate COCOMO Model**

The basic COCOMO model considers that effort and development time depends only on the size of the software. However, in real life there are many other project parameters that influence the development process. The intermediate COCOMO take those other factors into consideration by defining a set of 15 cost drivers (multipliers) as shown in the table below . Thus, any project that makes use of modern programming practices would have lower estimates in terms of effort and cost. Each of the 15 such attributes can be rated on a six-point scale ranging from "very low" to "extra high" in their relative order of importance. Each attribute has an effort multiplier fixed as per the rating. The product of effort multipliers of all the 15 attributes gives the **Effort Adjustment Factor (EAF).**





**Complete COCOMO Model**

Both the basic and intermediate COCOMO models consider a software to be a single homogeneous entity -- an assumption, which is rarely true. In fact, many real life applications are made up of several smaller sub-systems. (One might not even develop all the sub-systems -- just use the available services). The complete COCOMO model takes these factors into account to provide a far more accurate estimate of project metrics.

To illustrate this, consider a very popular distributed application: the ticket booking system of the Indian Railways. There are computerized ticket counters in most of the railway stations of our country. Tickets can be booked / cancelled from any such counter. Reservations for future tickets, cancellation of reserved tickets could also be performed.

On a high level, the ticket booking system has three main components:

* Database
* Graphical User Interface (GUI)
* Networking facilities
* Among these, development of the GUI is considered as an organic project type; the database module could be considered as a semi-detached software. The networking module can be considered as an embedded software. To obtain a realistic cost, one should estimate the costs for each component separately, and then add it up.

**A.5 Task to be completed in PART B**

**A.5.1. Task 1:**

**Every student needs to follow following steps and record the findings in appropriate section of PART B**

1. Identify the type of Project (Basic or Intermediate)

It is a Basic COCOMO Model ,where we will calculate the cost of project on a rough calculation basis with limited factors.

1. Compute Time to develop, Effort required for Complete COCOMO model (Considering one basic module, one semidetached module and one embedded module).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**PART B**

(PART B: TO BE COMPLETED BY STUDENTS)

**(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)**

|  |  |
| --- | --- |
| Roll No. B228 | Name: Pranav Kolhe |
| Program: Btech CS | Division: A |
| Batch: A | Date of Experiment: |
| Date of Submission: | Grade : |

**B.1 Tasks given in PART A to be completed here**

*(****Students must write the answers of the task(s) given in the PART A)***

1. Identify the type of Project (Basic or Intermediate)

It is a Basic COCOMO Model, where we will calculate the cost of project on a rough calculation basis with limited factors.

1. Compute Time to develop, Effort required for Complete COCOMO model (Considering one basic module, one semidetached module and one embedded module).

**We Know,**

**Basic COCOMO Model**

The basic COCOMO model helps to obtain a rough estimate of the project parameters. It estimates effort and time required for development in the following way:   
Text

Description automatically generated

Where,

* KDSI is the estimated size of the software expressed in Kilo Delivered Source Instructions.
* a, b, c are constants determined by the category of software project
* Effort denotes the total effort required for the software development, expressed in person months (PMs)
* Tdev denotes the estimated time required to develop the software (expressed in months)

the relative levels of product development complexity for the three categories (application, utility and

system programs) of products are 1:3:9.

Organic:

We can consider a development project to be of organic type, if the project

deals with developing a well understood application program, the size of the

development team is reasonably small, and the team members are experienced in

developing similar types of projects.

Semidetached:

The project can be considered of this type is the development team

consists of a mixture of experienced and inexperienced staff. Team members may

have limited experience on related systems but may be unfamiliar with some aspects

of the system being developed.

Embedded:

The project can be considered of this type, is the software being

developed is strongly coupled to complex hardware, or if stringent regulations on the

operational procedures exist. Here we have chosen the embedded model since all the

team members are new in the field of the software development and also the software

product size is relatively small.

Estimation of Development Efforts

1. Organic model:

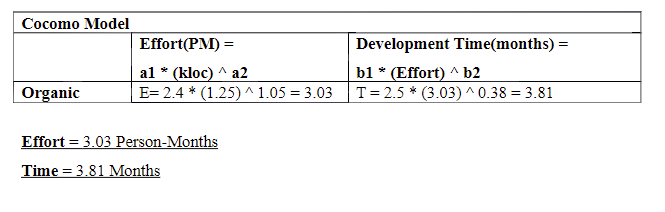
Efforts = 2.4(KLOC)1.05 Person-Months

1. Semi-Detached Model:

Efforts = 3.0(KLOC) 1.12 Person-Months

1. Embedded Model:

Efforts=3.6(KLOC) 1.20 Person-Months



**B.2 Observations and Learning:**

*(****Students must write the observations and learning based on their understanding built about the subject matter and inferences drawn)***

**COCOMO Model was learnt in this experiment.**

**B.3 Conclusion:**

*(****Students must write the conclusive statements as per the attainment of individual outcomes listed above and learning/observation noted in section B.2)***

**Experiment was executed successfully.**

**B.4 Question of curiosity:**

1. Identify the different estimation techniques followed in industry.

**1) Delphi Technique  
2) Work Breakdown Structure (WBS)  
3) Three Point Estimation  
4) Functional Point Method**

1. List down advantages and limitations of COCOMO model

**Advantages :**

* It works on historical data and provides more accurate details.
* Easy to implement with various factors. One can easily understand how it works.
* Easy to estimate the total cost of the project.
* The drivers are very helpful to understand the impact of the different factors that affect project crises.

**Disadvantages :**

* It ignores the hardware issues as well as the personal turnover level.
* It ignores all the documentation and requirements.
* It mostly depends on time factors.
* It limits the accuracy of software costs.
* It oversimplifies the impact of safety or security aspects.
* It also ignores customer skills, cooperation, and knowledge.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*