Batch: D-2	Roll No.: 16010123324 16010123325 16010123331

Experiment / assignment / tutorial No._3___

TITLE: Project Metric estimations for Mini Project

AIM: To enable the students learn different techniques for performing software size and cost estimation

Expected Course outcome of Experiment:

Books/ Journals/ Websites referred:

Roger Pressman, "Software Engineering", sixth edition, Tata McGraw Hill.
http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html
http://sunset.usc.edu/research/COCOMOII/expert_cocomo/expert_cocomo2000.html

Pre Lab/ Prior Concepts:

Software projects have tendency of going past their deadline, going over budget, or both. The problem lies in the estimation of the amount of effort required for the development of a project. The cost estimation is usually dependent upon the size estimate of the project, which may use lines of code or function points as metrics. There are several different techniques for performing software cost estimation, including expert judgement and algorithmic models. Estimation by expert judgement is a common way of estimating the effort required for a project. Unfortunately, this method of estimation does not emphasize re-estimation during the project life cycle, which is an important part of project tracking, because it allows the estimates to be improved during the project life cycle. The quality of a cost estimation model is not so much attributed to the initial estimate, but rather the speed at which the estimates converge to the actual cost of the project. COCOMO is a popular algorithmic model for cost estimation whose cost factors can be tailored to the individual development environment, which is important for the accuracy of the cost estimates. More than one method of cost estimation should be done so that there is some comparison available for the estimates. This is especially important for unique projects. Cost estimation must be done more diligently throughout the project

life cycle so that in the future there are fewer surprises and unforseen delays in the release of a product.

Estimation of size and cost of the developing project is required for the following major decision situations

Financial decisions involving a software development effort
Setting project budgets and schedules as a basis for planning and control
Deciding on or negotiating tradeoffs among software cost, schedule, functionality, performance or quality factors
Making software cost and schedule risk management decisions
Deciding which parts of a software system to develop, reuse, lease, or purchase
Making legacy software inventory decisions: what parts to modify, phase out, outsource, etc
Deciding how to implement a process improvement strategy, such as that provided in the SEI CMM

Defining Cost estimation:

Cost estimation can be defined as the approximate judgement of the costs for a project. Cost estimation will never be an exact science because there are too many variables involved in the calculation for a cost estimate, such as human, technical, environmental, and political. Futhermore, any process that involves a significant human factor can never be exact because humans are far too complex to be entirely predictable. Furthermore, software development for any fair-sized project will inevitably include a number of tasks that have complexities that are difficult to judge because of the complexity of software systems.

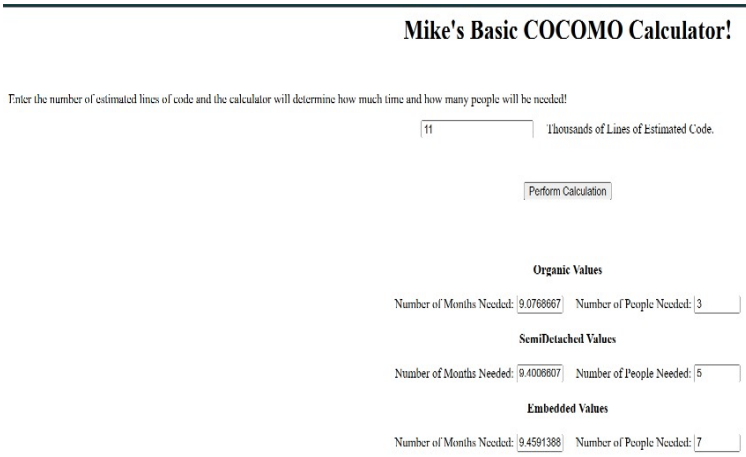
Cost estimation is usually measured in terms of effort. The most common metric used is person months or years (or man months or years). The effort is the amount of time for one person to work for a certain period of time. It is important that the specific characteristics of the development environment are taking into account when comparing the effort of two or more projects because no two development environments are the same. A clear example of differences in development environments are the amount of time people work in different countries; the typical workweek in North America is 40 hours per week, while in Europe the typical workweek is 35 hours per week. Thus, when comparing a project from North America with a project from Europe, a conversion factor would have to be used for an accurate comparison. Different variables can be used for cost estimation, which leads to a difficulty when comparing projects if standard models or tools are not used. For example, a cost estimate can include factors from management, development (e.g., training, quality assurance), and other areas specific to an organization.

Estimator:

The people who do the cost estimates could be either directly or indirectly responsible for the implementation for a project, such as a developer or manager, respectively. Someone who has knowledge of the organization and previous projects could use an analogy-based approach to compare the current project with previous projects, which is a common method of estimation for small organizations and small projects. The historical data is often limited to the memory of the estimator. In this case, the estimator would need to be experienced and would likely have been with the company for a while.

Some people believe it is better if the estimates are done by outsiders so that there is less chance of bias. It is true that people outside an organization will likely have to deal with fewer company politics than people within the organization. For example, the developer for a company may want to please the manager and so give an estimate that is overly-optimistic. The disadvantage of having an outside estimate is that the person would have less knowledge of the development environment, especially if the person is from outside the company. An empirical method of estimation would then be required, such as the Constructive Cost Model (COCOMO. Empirical methods of estimation can be used by all types of estimators. There may be some resistance to using an empirical method of estimation because there may be some question on whether a model could outperform an expert. People who are accurate estimators are rare in our experience, and so it is best to get the opinion of several people or tools.

Cost estimation using different COCOMO models:



Function Point Based

Information Domain Value	Count	Simple	Average	Complex	Weight Used	Total FP
External Inputs (EIs)	6	3	4	6	4	24
External Outputs (EOs)	5	4	5	7	5	25
External Inquiries (EQs)	5	3	4	6	4	20
Internal Logical Files (ILFs)	3	7	10	15	10	30
External Interface Files (EIFs)	4	5	7	10	7	28
Total FP						127

Factor	Value
Backup and Recovery	4
Data Communications	3
Distributed Processing	3
Performance Critical	3
Existing Operation Environment	2
Online Data Entry	4
Input transaction over multiple screens	2
Master file updated on-line	4
Information domain values complex	3
Internal Processing Complex	3
Code designed for reuse	3
Conversion / Installation in design	2
Multiple Installations	2
Applications designed for change	4
Total	42

TDI = 4 + 3 + 3 + 3 + 2 + 4 + 2 + 4 + 3 + 3 + 3 + 2 + 2 + 4 = 42

FP = count total X [0.65 + 0.01 X ($\sum Fi$)]
= 127 x [0.65 + 0.01 x 42] = 135

LOC Matrix

Function / Module	Estimated LOC
UI Module	1,500
Authentication	1,000
Chat Handling	1,200
File Sharing	800
Notification Service	600
Database Module	1,200
Encryption Module	800
Event Creation & Management	1,000
RSVP Handling	500
Event Search & Filter	800
Search Algorithm	1,000
Personalized Suggestions	1,000
Total	**11,400**

Conclusion:

In this experiment, we applied **LOC (Lines of Code)**, **Function Point Based (FPB)**, and **COCOMO** methods for software effort estimation. LOC and FPB gave basic measures of project size, while COCOMO provided a more detailed estimation of effort, cost, and schedule by considering project characteristics and cost drivers. Together, these methods highlight how different estimation techniques complement each other and help in accurate project planning and management.

Post Lab Descriptive Questions

Explain COCOMO II model.

COCOMO II (Constructive Cost Model II) is a modern software cost estimation model developed by Barry Boehm to improve upon the original COCOMO. It estimates **effort, cost, and schedule** of software projects by considering factors like project size (in KLOC or function points), scale drivers (e.g., team cohesion, process maturity), and cost drivers (e.g., reliability, complexity, experience). It has three sub-models **Application Composition, Early Design, and Post-Architecture** used at different stages of development. The core formula is:

$$PM = A \times (Size)^E \times \prod EM_i$$

where PM is person-months, E depends on scale factors, and EMi are effort multipliers. COCOMO II is widely used as it accounts for modern practices like reuse, prototyping, and iterative development.