COCOMO Model

Boehm's COCOMO model is one of the mostly used model commercially. The first version of the model is delivered in 1981 and COCOMO II is available now. COCOMO'81 is derived from the analysis of 63 software projects in 1981. Boehm proposed three levels of the model.

(a) The basic COCOMO'81 Model : It is a single valued, static model that computes software development effort (and costs) as a function of program size expressed in estimated lines of code (LOC).

(b) The Intermediate COCOMO'81 Model: It computes software development effort as a function of program size and set of 'Cost driver's that include subjective assessments of product, hardware, personnel and project attributes.

(c) The detailed COCOMO'81 Model : It incorporates all charactersitics of the intermediate version with an assessment of the cost driver's impact on each step of the software engineering process.

COCOMO'81 models depends on the two main equations :

(a) First is development effort, is one month of effort by one person. In COCOMO'81, there are 152 hours per person-month. According to organization this values may differ from the standard by 10% to 20% for the basic model :

$$MM = a * KDSI * b$$

(b) Second is effort and developmentd time (TDEV)

$$TDEV = c * MM * d$$

KDSI means the number of thousand delivered source instructions and it is a measure of size. The coefficients a, b, c and d are dependent on the model of development. There are 3 months of development.

Development Mode	Projects Characteristics			
	Size	Innovation	Constraints	Development Environment
Organic	Small	Little	Not Tight	Stable
Semi-detached	Medium	Medium	Medium	Medium
Embedded	Large	Greater	Tight	Complex Hardware

Here are the coefficients related to development modes for intermediates model.

Development Mode	*a*	*b*	*c*	*d*
Organic	3·2	1·05	2·5	0·38
Semi-detached	3·0	1·12	2·5	0·35
Embedded	2·8	1·20	2·5	0·32

In intermediate mode development effort operations becomes

$$MM = a * KDSI * b * c$$

C (effort adjustment factor) is calculated simply multiplying the values of cost drivers. So, the intermediate model is more accurate than the basic model.

The steps in producing an estimating using the intermediate COCOMO'81 are :

(1) Identify the mode of development for the new product.
(2) Estimate the size of project in KDSI to derive a nominal effort prediction.
(3) Adjust 15 cost derivers to reflect your project.
(4) Calculate the predicted project effort using first equation is the adjustment factor(c).
(5) Calculate the project duration using second equation.

Advantages of COCOMO'81

(1) COCOMO is transparent, we can see how it works unlike other models such as SLIM.
(2) Drivers are particularly helpful to the estimator to understand the impact of different factors that affect project costs.

Disadvantages of COCOMO'81

(1) It is hard to accurately estimate KDSI early on in the project, when most effort estimates are required.
(2) KDSI actually is not a size measure, it is a length measure.

✔ Planning of Software Project

After the costs have been approved for the project, the first step in the execution of the project is preparing the project plan. Project planning involves all aspects relating to the project. It is the end document which can be used as a reference to know how various activities are to be performed. When they are to be taken up their dependencies and the effort required for each activity. Typically project planning involves the following activities :

(a) Methodology
(b) Risks
(c) Quality Plan
(d) Configuration Management Plan
(e) Project Schedule
(f) Resource Plan

(a) Methodology : Methodology is nothing but a series of steps which need to be carried out to meet an end objective. If software development, there are number of methodologies which can be followed for development of a software product. The popular methodologies used for software development are:

(i) Waterfall Model
(ii) Prototyping
(iii) Rapid Application Development

(b) Risks : Software projects have a number of risks associated with them. The risks described here relate to issues which could potentially affect the delivery or budgets of the projects. The main objectives of risk analysis is to identify all possible risks and document a plan to counter the risk in case of the unlikely event occurring. When a undesirable event occurs, it might be too late to counter the consequences unless adequate planning has been done to counter the risk. This is best understood by

✔ Project Size Estimation

Through the single variable cost models with size as the independent variable result in simple - looking models that can be easily obtained from completed projects (for which accurate data about sizes and cost are known), applying them for estimation is not simple. The reason is that these models now require size as the input, and size of the project is not known early in the development (when the estimates are most needed) and has to be estimated. This is particularly true if LOC is used as the measure of size in the model.

So, in a sense, by using size as the main input to the cost model, we have replaced the problem of the cost estimation by size estimation. One may then ask, why not directly do cost estimation rather than the size estimation? Because size estimation is generally easier than effort estimation. For estimating size, the system is generally partitioned into components it is likely to have once the components of the system are known, as estimating something about a small unit is generally much easier than estimating it for a larger system, sizes of components can be generally estimated quite accurately. Once size estimates for components are valuable, to get the overall size estimates for the system, the estimates of all the components can be added up. Similar property does not hold for cost estimation, as cost of developing a system is not the sum of costs of developing the components (there are integration and other costs involved when building a system from developed components).

This key feature, that the system property is the sum of the properties of its parts, holds for size but not for cost and is the main reason that size estimation is easier than cost estimation. For this reason, most cost estimation models, use size estimates in some form. When estimating software size, the best way may be to get as much detail as possible about the software to be developed and to be aware of our biases when estimating the size of various components. By obtaining details and using them for size estimation, the estimates are likley to be closer to the actual size of final software.