SOFTWARE ESTIMATION TECHNIQUE

Barry Boehm, in his classic work on software effort models, identified the main ways of deriving estimates of software development effort as:

1) **Algorithmic Models**

- which use 'effort drivers' representing characteristics of the target system and the implementation environment to predict effort:

2) **Expert Judgment**

- Where the advice of knowledgeable staff is solicited:

3) **Analogy**

- Here a similar, completed, project is identified and its actual effort is used as a basis for the new project;

4) **Parkinson**

- Which identities the staff effort available to do a project and uses that as the 'estimate';

5) **Price to Win**

- Where the 'estimate' is at figure that appears to be sufficiently low to win a contract:

6) **Top-Down**

- Where an overall estimate is formulated for the whole project and is then broken down into the effort required for component tasks:

7) **Bottom-Up**

- Where component tasks are identified and sited and these individual estimates are aggregated.

8) **Bottom-Up Estimating**

- Bottom-up approach, the estimator breaks the project into its component tasks and then estimates how much effort will be required to carry out each task.

- With a large project, the process of breaking down into tasks would be a repetitive one: each task would

be analyzed into its component sub-tasks and these in turn would be further analyzed.

9) The Top-Down Approach And Parametric Models

- The Top-Down Approach is normally associated with parametric models.
- The effort needed to implement a project will be related mainly to variables associated with characteristics of the final system.
- The form of the parametric model will normally be one or more formulae in the form

Effort = (system size) x (productivity rate)

- A model to forecast software development effort therefore has two key components.
- The first is a method of assessing the size of the software development task to be undertaken.
- The second assesses the rate of work at which the task can be done.
- Some parametric models such as that implied by function points, are focused on system or task size, while others, such are COCOMO, are more concerned with productivity factors

EXPERT JUDGMENT

- This is asking someone who is knowledgeable about either the application area or the development environment to give an estimate of the effort needed to carry out a task.
- This method will most likely be used when estimating the effort needed to change an existing piece of software.
- The estimator would have to carry out some kind of impact analysis in order to judge the proportion of

code that would be affected and from that derives an estimate.

- Someone already familiar with the software would be in the best position to do this.

ESTIMATING BY ANALOGY

- The use of analogy is also called **Case-Based Reasoning.**
- The estimator seeks out projects that have been completed and that have similar characteristics to the new project.
- The effort that has been recorded for the matching source case can then be used as a base estimate for the target.
- The estimator should then try to identify any differences between the target and the source and make adjustments to the base estimate for the new project.
- A problem here is how you actually identify the similarities and differences between the different systems.
- Attempts have been made to automate this process.
- One software application that has been developed to do this is ANGEL.
- This identifies the source case that is nearest the target by measuring the Euclidean distance between cases.
- The source case that is at the shortest Euclidean distance from the target is deemed to be the closest

  match.
- The Euclidean distance is calculated:

Euclidean distance = square-root of (target_parameter$_1$, Source_parameter$_1$)^2 + + (target_parameter$_n$, Source_parameter$_n$)

…

Function Point Analysis

- The basis of function point analysis is that computer-based information systems comprise five major components, or external users type that are of benefit to the users:

  1) **External** input types are input transactions that update internal computer tiles.

2) **External** output types are transactions where data is output to the user. Typically these would he printed reports, since screen displays would come under external inquiry types

3) **Logical** internal tile **types** are the standing files used by the system. The term 'file does not sit easily with modern information systems. It refers to a group of data that is usually accessed together. It might be made up of one or more *record types.*

4) **External interface *file* types** allow for output and input that might pass to and from other computer applications. Files shared among applications would also be counted here.

5) **External inquiry types -** note *the* US spelling of inquiry - are transactions initiated by the user that provide information but do not update *the* internal tiles

The user inputs some information that directs the system to the details required

- The analyst has to identify each instance of each external user type in the projected system.
- Each component is then classified as having high, average or low complexity.
- The counts of each external user type in each complexity hand arc multiplied by specified weights to get

IT scores, which are summed to obtain an overall FP count, which indicates the information processing size.

- Estimates are really management targets;
- Collect as much information about previous project as possible.
- Top-down approaches will be used at the earlier stages of project planning while bottom-up approaches will be on later stages