Empirical Estimation Models

- Estimation models for computer software use empirically derived formulas to predict effort as a function of LOC (line of code) or FP(function point)
- Resultant values computed for LOC or FP are entered into an estimation model
- The empirical data for these models are derived from a limited sample of projects
  - Consequently, the models should be calibrated to reflect local software development conditions

$$E = A + B \times (e_v)^C \qquad (26.3)$$

where A, B, and C are empirically derived constants, E is effort in person-months, and e_v is the estimation variable (either LOC or FP

many LOC-oriented estimation models proposed in the literature are

$E = 5.2 \times (\text{KLOC})^{0.91}$	Walston-Felix model
$E = 5.5 + 0.73 \times (\text{KLOC})^{1.16}$	Bailey-Basili model
$E = 3.2 \times (\text{KLOC})^{1.05}$	Boehm simple model
$E = 5.288 \times (\text{KLOC})^{1.047}$	Doty model for KLOC > 9

FP-oriented models have also been proposed. These include

$E = -91.4 + 0.355\ \text{FP}$	Albrecht and Gaffney model
$E = -37 + 0.96\ \text{FP}$	Kemerer model
$E = -12.88 + 0.405\ \text{FP}$	Small project regression model

COCOMO

- Stands for Constructive Cost Model
- Introduced by Barry Boehm in 1981 in his book "Software Engineering Economics"
- Became one of the well-known and widely-used estimation models in the industry
- It has evolved into a more comprehensive estimation model called COCOMO II
- COCOMO II is actually a hierarchy of three estimation models
- As with all estimation models, it requires sizing information and accepts it in three forms: object points, function points, and lines of source code

(More on next slide)

COCOMO –II

- uses object points
- *object point* *is an indirect software measure that is computed* using counts of the number of (1) screens (at the user interface), (2) reports, and (3) components likely to be required to build the application.
- Each object instance (e.g., a screen or report) is classified into one of three complexity levels (i.e., simple, medium, or difficult) using criteria suggested by Boehm
- Once complexity is determined, the number of screens, reports, and components are weighted according to the table illustrated
- The object point count is then determined by multiplying the original number of object instances by the weighting factor and summing to obtain a total object point count

Number of views contained	*# and sources of data tables*		
	Total < 4 (< 2 server < 3 client)	*Total < 8 (2 – 3 server 3 – 5 client)*	*Total 8 + (> 3 server, > 5 client)*
< 3	Simple	Simple	Medium
3 – 7	Simple	Medium	Difficult
> 8	Medium	Difficult	Difficult

Table 9 (a): For screens

Number of sections contained	*# and sources of data tables*		
	Total < 4 (< 2 server < 3 client)	*Total < 8 (2 – 3 server 3 – 5 client)*	*Total 8 + (> 3 server, > 5 client)*
0 or 1	Simple	Simple	Medium
2 or 3	Simple	Medium	Difficult
4 +	Medium	Difficult	Difficult

Table 9 (b): For reports

Object Type	*Complexity Weight*		
	Simple	*Medium*	*Difficult*
Screen	1	2	3
Report	2	5	8
3GL Component	—	—	10

Table 10: Complexity weights for each level

COCOMO -II

When the software is reused ...then percentage of reuse is estimated and the object point count is adjusted by using the formula

- NOP = (object points) X [(100 - %reuse)/100]

where NOP is defined as new object points.

- To derive an estimate of effort based on the computed NOP value, a "productivity rate" must be derived.

$$PROD = \frac{NOP}{person\text{-}month}$$

Developers experience & capability	Productivity (PROD)
Very Low	4
Low	7
Nominal	13
High	25
High	50

Productivity Rate

- Once the productivity rate has been determined, an estimate of project effort.

$$\text{Estimated effort} = \frac{NOP}{PROD}$$

Example

- https://www.geeksforgeeks.org/software-engineering-application-composition-estimation-model-cocomo-ii-stage-1/
- Consider a database application project with The application has four screens with four views each and seven data tables for three servers and four clients. Application may generate two reports of six section each from seven data tables for two servers and three clients. 10% reuse of object points.
  Developer's experience and capability in similar environment is low. Calculate the object point count, New object point and effort to develop such project.