COCOMO - An Empirical Estimation Model for Effort

Introduction:

The structure of empirical estimation models is a formula, derived from data collected from past software projects, that uses software size to estimate effort. Size, itself, is an estimate, described as either lines of code (LOC) or function points (FP). No estimation model is appropriate for all development environments, development processes, or application types. Models must be customised (values in the formula must be altered) so that results from the model agree with the data from the particular environment.

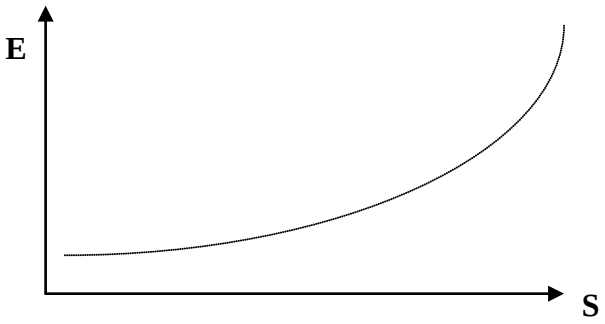
The typical formula of estimation models is:

$$\mathbf{E = a + b(S)^c}$$

where;

E represents effort, in person months,
S is the size of the software development, in LOC or FP, and,
a, **b**, and **c** are values derived from data.

The relationship seen between development effort and software size is generally:



This graph demonstrates that the amount of effort accelerates as size increases, i.e., the value **c** in the typical formula above is greater than 1.

COCOMO:

When Barry Boehm wrote 'Software Engineering Economics', published in 1981, he introduced an empirical effort estimation model (COCOMO - *COnstructive COst MOdel*) that is still referenced by the software engineering community. The model has been reviewed since 1981 and details of the revised and updated COCOMO 2 model, can be obtained at:
http://sunset.usc.edu/research/index.html .

The original COCOMO model was a set of models; 3 development modes (organic, semi-detached, and embedded) and 3 levels (basic, intermediate, and advanced).

COCOMO model levels:

Basic - predicted software size (lines of code) was used to estimate development effort.
Intermediate - predicted software size (lines of code), plus a set of 15 subjectively assessed 'cost drivers' was used to estimate development effort.
Advanced - on top of the intermediate model, the advanced model allows phase-based cost driver adjustments and some adjustments at the module, component, and system level.

COCOMO development modes:

Organic - small relatively small, simple software projects in which small teams with good application experience work to a set of flexible requirements.
Embedded - the software project has tight software, hardware and operational constraints.
Semi-detached - an intermediate (in size and complexity) software project in which teams with mixed experience levels must meet a mix of rigid and less than rigid requirements.[1]

COCOMO model:

The general formula of the basic COCOMO model is:

$$\mathbf{E = a(S)^b}$$

Where:

E represents effort in person-months,
S is the size of the software development in KLOC and,
a and **b** are values dependent on the development mode,

development mode:	organic	**a** = 2.4	**b** = 1.05
	semi-detached	**a** = 3.0	**b** = 1.12
	embedded	**a** = 3.6	**b** = 1.20

The intermediate and advanced COCOMO models incorporate 15 'cost drivers'. These 'drivers' multiply the effort derived for the basic COCOMO model. The importance of each driver is assessed and the corresponding value multiplied into the COCOMO equation, which becomes:

$$\mathbf{E = a(S)^b \times product(cost\ drivers)}$$

As an example of how the intermediate COCOMO model works, the following is a calculation of the estimated effort for a semi-detached project of 56 KLOC. The cost drivers are set as follows:

Product cost drivers (from the table) set	**high**	= 1.15 x 1.08 x 1.15	= 1.43
Computer cost drivers (from the table) set	**nominal**	= 1.00	
Personnel cost drivers (from the table) set	**low**	= 1.19 x 1.13 x 1.17 x 1.10 x 1.07	= 1.85
Project cost drivers (from the table) set	**high**	= 0.91 x 0.91 x 1.04	= 0.86

hence, **product(cost drivers)** = 1.43 x 1.00 x 1.85 x 0.86 = 2.28

for a semi-detached project of 56KLOC: **a** = 3.0 **b** = 1.12 **S** = 56

$$\mathbf{E = a(S)b \times product(cost\ drivers)}$$

$$\mathbf{E} = 3.0 \times (56)^{1.12} \times 2.28$$

$$\mathbf{E} = 3.0 \times 90.78 \times 2.28$$

$$\mathbf{E} = 620.94 \text{ person-months}$$

[1] *COCOMO development mode definitions are taken from: Software Engineering 4th Ed., Pressman RS, McGraw Hill.*

Description, and Table of values, for COCOMO Cost Drivers:

COST DRIVER	DESCRIPTION
RELY	Required software reliability
DATA	Database size
CPLX	Product complexity
TIME	Execution time constraints
STOR	Main storage constraints
VIRT	Virtual machine volatility - degree to which the operating system changes
TURN	Computer turn around time
ACAP	Analyst capability
AEXP	Application experience
PCAP	Programmer capability
VEXP	Virtual machine (i.e. operating system) experience
LEXP	Programming language experience
MODP	Use of modern programming practices
TOOL	Use of software tools
SCED	Required development schedule

COCOMO - COST DRIVERS

		RATING					
	COST DRIVER	**V.LOW**	**LOW**	**NOMINAL**	**HIGH**	**V.HIGH**	**EX. HIGH**
(PRODUCT)	**RELY**	**0.75**	**0.88**	**1.00**	**1.15**	**1.40**	**.**
..	**DATA**	**.**	**0.94**	**1.00**	**1.08**	**1.16**	**.**
..	**CPLX**	**0.70**	**0.85**	**1.00**	**1.15**	**1.30**	**1.65**
(COMPUTER)	**TIME**	**.**	**.**	**1.00**	**1.11**	**1.30**	**1.66**
..	**STOR**	**.**	**.**	**1.00**	**1.06**	**1.21**	**1.56**
..	**VIRT**	**.**	**0.87**	**1.00**	**1.15**	**1.30**	**.**
..	**TURN**	**.**	**0.87**	**1.00**	**1.07**	**1.15**	**.**
(PERSONNEL)	**ACAP**	**1.46**	**1.19**	**1.00**	**0.86**	**0.71**	**.**
..	**AEXP**	**1.29**	**1.13**	**1.00**	**0.91**	**0.82**	**.**
..	**PCAP**	**1.42**	**1.17**	**1.00**	**0.86**	**0.70**	**.**
..	**VEXP**	**1.21**	**1.10**	**1.00**	**0.90**	**.**	**.**
..	**LEXP**	**1.14**	**1.07**	**1.00**	**0.95**	**.**	**.**
(PROJECT)	**MODP**	**1.24**	**1.10**	**1.00**	**0.91**	**0.82**	**.**
..	**TOOL**	**1.24**	**1.10**	**1.00**	**0.91**	**0.83**	**.**
..	**SCED**	**1.23**	**1.08**	**1.00**	**1.04**	**1.10**	**.**

Questions:

1. For the above cost drivers to be multiplied together the underlying assumption is that they must be independent of each other, does this sound reasonable?
2. If all cost drivers were at minimum value, what would be the product of the cost drivers?
3. If all cost drivers were at maximum value, what would be the product of the cost drivers?
4. Based on the results to the two questions above, what is the ratio between maximum and minimum possible predicted effort?
5. Does the possible range from maximum to minimum effort seem reasonable?
6. The **SCED** cost driver is unique in the above table - can you see, and explain, why?
7. **COCOMO** starts from estimate of size, subjective assessment of 15 (independent?) cost drivers (with values - on potentially a 6-point scale - based on 60+ datasets), to estimate effort. How much confidence should you have in the final estimate?

Final Observations:

- An effort estimation is only, ever, an estimate. Management should treat it with caution.
- To make empirical models as useful as possible, as much data as possible should be collected from projects and used to customise (refine) any model used. An ongoing data collection programme is essential if models are to developed and refined, and if management wishes to make informed decisions
- Many organisations are known to use 'estimation redundancy', i.e., to provide a check on a particular estimate they will use more than one estimating method. The result is usually a set of estimates from which an organisation will choose. The underlying assumptions play a large role in the final decision and organisation makes.
- The different estimating methods used should be documented, and all underlying assumptions should be recorded.