COCOMO

COCOMO (Constructive Cost Model) is a model that allows software project managers to estimate project cost and duration. It was developed initially (COCOMO 81) by Barry Boehm in the early eighties. The COCOMO II model is a COCOMO 81 update to address software development practices in the 1990's and 2000's. The model is by now invigorative software engineering artifact that has, from customer perspective, the following features:

1. The model is simple and well tested
2. Provides about 20% cost and 70% time estimate accuracy

COCOMO 81(Basic)

1. It is an open system First published by Dr Barry Boehm in 1981
2. Worked quite well for projects in the 80's and early 90's
3. Could estimate results within ~20% of the actual values 68% of the time
4. COCOMO has three different modes:

   Organic – relatively small software teams develop software in a highly familiar, in-house environment

   Embedded – operate within tight constraints, product is strongly tied to complex of hardware, software, regulations, and operational procedures

   Semi-detached – an intermediate project in which mixed teams must work to a set of rigid and less than rigid requirements (ie. a transaction processing system with fixed requirements for terminal hardware and software).

5. COCOMO has three different models (each one increasing with detail and accuracy):

   Basic: It is applied early in a project. The Basic COCOMO model computes effort as function of program size.

   The Basic COCOMO equation is:

   $$E = a \times KLOC^b$$

Mode	Basic		Intermediate	
	a	b	a	b
Organic	2.4	1.05	3.2	1.05
Semi-detached	3.0	1.12	3.0	1.12
Embedded	3.6	1.20	2.8	1.20

Intermediate: It is applied after requirements are specified. The Intermediate COCOMO model computes effort as a function of program size and a set of cost drivers.

The Intermediate COCOMO equation is:

E = a x KLOC^b x EAF

The factors a and b are shown in the following table.

The effort adjustment factor (EAF) is calculated using 15 cost drivers. The cost drivers are grouped into four categories: *product*, *computer*, *personnel*, and *project*. Each cost driver is rated on a six-point ordinal scale ranging from low to high importance. Based on the rating, an effort multiplier is determined using the table below. The product of all effort multipliers is the EAF.

Software Development Effort Multipliers:

Cost Driver	**Description**	**Rating**					
		Very Low	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
Product							
RELY	Required software reliability	0.75	0.88	1.00	1.15	1.40	-
DATA	Database size	-	0.94	1.00	1.08	1.16	-
CPLX	Product complexity	0.70	0.85	1.00	1.15	1.30	1.65
Computer							
TIME	Execution time constraint	-	-	1.00	1.11	1.30	1.66

STOR	Main storage constraint	-	-	1.00	1.06	1.21	1.56
VIRT	Virtual machine volatility	-	0.87	1.00	1.15	1.30	-
TURN	Computer turnaround time	-	0.87	1.00	1.07	1.15	-
Personnel							
ACAP	Analyst capability	1.46	1.19	1.00	0.86	0.71	-
AEXP	Applications experience	1.29	1.13	1.00	0.91	0.82	-
PCAP	Programmer capability	1.42	1.17	1.00	0.86	0.70	-
VEXP	Virtual m/c experience	1.21	1.10	1.00	0.90	-	-
LEXP	Language experience	1.14	1.07	1.00	0.95	-	-
Project							
MODP	Modern prg practices	1.24	1.10	1.00	0.91	0.82	-
TOOL	Software Tools	1.24	1.10	1.00	0.91	0.83	-
SCED	Development Schedule	1.23	1.08	1.00	1.04	1.10	-

Advanced*:* It is applied after the design is complete. The Advanced COCOMO model computes effort as a function of program size and a set of cost drivers weighted according to each phase of the software lifecycle. The Advanced model applies the Intermediate model at the component level, and then a phase-based approach is used to consolidate the estimate.

The 4 phases used in the detailed COCOMO model are: requirements planning and product design (RPD), detailed design (DD), code and unit test (CUT), and integration and test (IT). Each cost driver is broken down by phase as in the example shown in the following table.

Analyst capability effort multiplier for Detailed COCOMO:

Cost Driver	Rating	RPD	DD	CUT	IT
ACAP	Very Low	1.80	1.35	1.35	1.50
	Low	0.85	0.85	0.85	1.20
	Nominal	1.00	1.00	1.00	1.00
	High	0.75	0.90	0.90	0.85
	Very High	0.55	0.75	0.75	0.70

Estimates made for each module are combined into subsystems and eventually an overall project estimate. Using the detailed cost drivers, an estimate is determined for each phase of the lifecycle.

Advantages

1. COCOMO is transparent, you 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

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.
3. Extremely vulnerable to mis-classification of the development mode
4. Success depends largely on tuning the model to the needs of the organization, using historical data which is not always available

COCOMO II

1. Whereas COCOMO is reasonably well matched to custom, build-to-specification software projects, COCOMO II is useful for a much wider collection of techniques and technologies.
2. COCOMO II provides up-to-date support for business software, object-oriented software, software created via spiral or evolutionary development models, and software developed using commercial-off-the-shelf application composition utilities
3. COCOMO II has three different models:

Application Composition model: The Application Composition model is used in prototyping to resolve potential high-risk issues such as user interfaces, software/system interaction, performance, or technology maturity. Object points are used for sizing rather than the traditional LOC metric. An initial size measure is determined by counting the number of screens, reports, and third-generation components that will be used in the application. Each object is classified as simple, medium, or difficult.

Object point complexity levels for screens and reports.

Number of views contained	Number and source of data tables		
	Total <4	**Total <8**	**Total 8+**
<3	simple	simple	Medium
3-7	simple	medium	difficult
8+	medium	difficult	difficult

The number in each cell is then weighted according to the following table. The weights represent the relative effort required to implement an instance of that complexity level.

Complexity weights for object points

Object type	Simple	Medium	Difficult
Screen	1	2	3
Report	2	5	8
3GL component	-	-	10

The weighted instances are summed to provide a single object point number. Reuse is then taken into account. Assuming that *r%* of the objects will be reused from previous projects; the number of new object points (NOP) is calculated to be:

NOP = (object points) x (100 – r) / 100

A productivity rate (PROD) is determined using the following table.

Average productivity rates based on developers:

Developers' experience and capability	Very Low	Low	Nominal	High	Very High
PROD	4	7	13	25	50

PROD = __________ nop/person-months

Effort can then be estimated using the following equation:

E = NOP / PROD person-months

Early Design model: The Early Design model is used to evaluate alternative software/ system architectures and concepts of operation. An unadjusted function point count (UFC) is used for sizing. This value is converted to LOC using following tables.

Programming language levels and ranges of source code statements per function point:

Language	Level	Min	Mode	Max
Machine language	0.10	-	640	-
Assembly	1.00	237	320	416
C	2.50	60	128	170
RPGII	5.50	40	58	85
C++	6.00	40	55	140
Visual C++	9.50	-	34	-
PowerBuilder	20.00	-	16	-
Excel	57.00	-	5.5	-

The Early Design model equation is:

E = a x KLOC x EAF

where *a* is a constant, provisionally set to 2.45.

The effort adjustment factor (EAF) is calculated as in the original COCOMO model using the 7 cost drivers shown in the table.

Early Design cost drivers.

Cost Driver	Description	Counterpart Combined Post-Architecture Cost Driver
RCPX	Product reliability and complexity	RELY, DATA, CPLX, DOCU
RUSE	Required reuse	RUSE
PDIF	Platform difficulty	TIME, STOR, PVOL
PERS	Personnel capability	ACAP, PCAP, PCON

PREX	Personnel experience	AEXP, PEXP, LTEX
FCIL	Facilities	TOOL, SITE
SCED	Schedule	SCED

Post Architecture model: is used during the actual development and maintenance of a product. Function points or LOC can be used for sizing, with modifiers for reuse and software breakage. The model includes a set of 17 cost drivers and a set of 5 factors determining the projects scaling component. The 5 factors replace the development modes (organic, semidetached, and embedded) of the original COCOMO model.

The Post-Architecture model equation is:

E = a x KLOC^b x EAF

Where *a* is set to 2.55 and *b* is calculated as:

b = 1.01 + 0.01 x ∑ (Wi)

Where *W* is the set of 5 scale factors shown in table below

COCOMO II scale factors.

W(i)	Very Low	Low	Nominal	High	Very High	Extra High
Precedence	4.05	3.24	2.42	1.62	0.81	0.00
Development/ Flexibility	6.07	4.86	3.64	2.43	1.21	0.00
Architecture / Risk Resolution	4.22	3.38	2.53	1.69	0.84	0.00
Team Cohesion	4.94	3.95	2.97	1.98	0.99	0.00
Process Maturity	4.54	3.64	2.73	1.82	0.91	0.00

The EAF is calculated using the 17 cost drivers shown in following table.

Post-Architecture cost drivers.

	Description	Rating

Cost Driver		**Very Low**	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
Product							
RELY	Required software reliability	0.75	0.88	1.00	1.15	1.39	-
DATA	Database size	-	0.93	1.00	1.09	1.19	-
CPLX	Product complexity	0.70	0.88	1.00	1.15	1.30	1.66
RUSE	Required reusability		0.91	1.00	1.14	1.29	1.49
DOCU	Documentation		0.95	1.00	1.06	1.13	
Platform							
TIME	Execution time constraint	-	-	1.00	1.11	1.31	1.67
STOR	Main storage constraint	-	-	1.00	1.06	1.21	1.57
PVOL	Platform volatility	-	0.87	1.00	1.15	1.30	-
Personnel							
ACAP	Analyst capability	1.50	1.22	1.00	0.83	0.67	-
PCAP	Programmer capability	1.37	1.16	1.00	0.87	0.74	-
PCON	Personnel continuity	1.24	1.10	1.00	0.92	0.84	-
AEXP	Applications experience	1.22	1.10	1.00	0.89	0.81	-
PEXP	Platform experience	1.25	1.12	1.00	0.88	0.81	-
LTEX	Language and tool experience	1.22	1.10	1.00	0.91	0.84	
Project							
TOOL	Software Tools	1.24	1.12	1.00	0.86	0.72	-

SITE	Multisite development	1.25	1.10	1.00	0.92	0.84	0.78
SCED	Development Schedule	1.29	1.10	1.00	1.00	1.00	-