COCOMO-I

SPARSH ACHARYA
23FE10CAI00367

Overview

COCOMO stands for the **Constructive Cost Model** and was developed by Barry Boehm in 1981. It is an empirical algorithmic model that estimates software development effort, schedule, and staffing based on the estimated size of the software (typically expressed in KLOC—thousands of lines of code) and other project attributes.

Project Types

COCOMO 1 distinguishes three project types based on size, complexity, environment, and team experience. These types help adjust the estimation coefficients.
They are:

- Organic Projects
- semi-Detached Projects
- Embedded Projects

Organic Project

- **Characteristics:**
  - Small, relatively simple projects.
  - Teams are small and experienced.
  - Problems are well understood.
  - Requirements are stable and less rigid.
- **KLOC Range:** Approximately 2 to 50 KLOC.
- **Examples:**
  - Simple business applications, basic inventory systems, or data processing systems.

Semi-Detached Project

- **Characteristics:**
  - Intermediate complexity and size.
  - Teams are mixed in experience.
  - Requirements may be partially rigid and may require moderate innovation.
- **KLOC Range:** Approximately 50 to 300 KLOC.
- **Examples:**
  - Database management systems, transaction processing systems, or moderately complex inventory applications.

Embedded Projects

- **Characteristics:**
  - High complexity with stringent constraints (hardware, software, real-time requirements).
  - Larger teams often required.
  - Often involve safety-critical or mission-critical systems.
- **KLOC Range:** Typically more than 300 KLOC.
- **Examples:**
  - Air traffic control systems, real-time embedded control systems (such as ATMs or automotive control systems).

Estimation Models

COCOMO 1 comes in three “levels” or forms which offer progressively more detailed estimation:

- Basic
- Intermediate
- Detailed

Basic Model

The Basic COCOMO model estimates effort solely as a function of the project size (KLOC) using a simple power-law equation:

- **Efforts:**

$$E = a \times (KLOC)^b \quad PM$$

- **Development Time:**

$$D = c \times E^d \quad M$$

- **Average Staff Size:**

$$staff = {}^{E}/_{D} \quad P$$

- **Productivity:**

$$Prod = {}^{KLOC}/_{E}$$

Coeff of Basic Model

Project Type	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Numerical 1

(b) For a given project was estimated with a size of 200 KLOC. Calculate the Effort, Scheduled time for development. Also, calculate the Average resource size and Productivity of the software for Embedded project type.

Project Type	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semidetached	3	1.12	2.5	0.35
Embedded	3.6	1.2	2.5	0.32

Solution

- Given
  KLOC = 200
  Proj Type = Embedded

  so a = 3.6, b = 1.2, c = 2.5, d = 0.32

  substituting in equations
- **Efforts:**

$$E = a \times (KLOC)^b = 3.6 \times (200)^{1.2} = 2077.48\ PM$$

- **Development Time:**

$$D = c \times E^d = 2.5 \times 2077.48^{0.32} = 28.81\ M$$

Solution

- **Average Staff Size:**

$$staff = {}^{E}/_{D} = {}^{2077.48}/_{28.81} = 72.11\ P$$

- **Productivity:**

$$Prod = {}^{KLOC}/_{E} = {}^{200}/_{2077.48} = 0.096\ {}^{KLOC}/_{PM}$$

Intermediate COCOMO Model

Intermediate COCOMO extends the basic model by incorporating 15 cost drivers (also called "effort multipliers") that adjust the nominal effort estimate. These drivers reflect factors related to the product, hardware, personnel, and project attributes.

Efforts:

$$E = a \times (KLOC)^b \times EAF \quad PM$$

Coeff of Intermediate Model

Project Type	a	b	c	d
Organic	3.2	1.05	2.5	0.38
Semi-Organic	3.0	1.12	2.5	0.35
Embedded	2.8	1.20	2.5	0.32

Cost Drivers

There are 15 cost drivers , divided into four categories:

- Product Attribute
- Hardware Attribute
- Personal Attribute
- Project Attribute

Each cost driver is rated (e.g., Very Low, Low, Nominal, High, Very High, Extra High) and has a corresponding multiplier. The product of all these multipliers yields the EAF, which typically ranges from about 0.9 to 1.4.

Product Attributes:

- Required software reliability (RELY)
- Database size (DATA)
- Product complexity (CPLX)

Hardware Attributes:

- Execution time constraints (TIME)
- Main storage constraints (STOR)
- Virtual machine volatility (VIRT)
- Turnaround time (TURN)

Personnel Attributes:

- Analyst capability (ACAP)
- Software engineering capability (AEXP/PCAP)
- Programmer capability (PCAP)
- Virtual machine experience (VEXP)
- Programming language experience (LEXP)

Project Attributes:

- Use of modern programming practices (MODP)
- Use of software tools (TOOL)
- Required development schedule (SCED)

Cost Driver	Very Low	Low	Nominal	High	Veri High	Extra High
RELY	0.75	0.88	1	1.15	1.40	-
DATA	-	0.94	1	1.08	1.16	-
CPLX	0.7	0.85	1	1.15	1.30	1.65
TIME	-	-	1	1.11	1.30	1.66
STOR	-	-	1	1.06	1.21	1.56
VIRT	-	0.87	1	1.15	1.30	-
TURN	-	0.87	1	1.07	1.15	-
ACAP	1.46	1.19	1	0.86	0.71	-
AEXP	1.29	1.13	1	0.91	0.82	-
PCAP	1.42	1.17	1	0.86	0.70	-
VEXP	1.21	1.10	1	0.9	-	-
LEXP	1.14	1.07	1	0.95	-	-
MODP	1.24	1.10	1	0.91	0.82	-
TOOL	1.29	1.10	1	0.91	0.83	-
SCED	1.24	1.10	1	1.04	1.1	-

Numerical 2

A new project with estimated 400 KLOC embedded system has to be developed project manager has a choice of hiring from 2 pools of developers
1) very high application experience with little experience in programming language

2) Developers of low App experience but a High programming language experience
which is better choice in terms of two pools?

Solution

- Given

  Cost Drivers:-

  - AEXP
  - LEXP

- Case 1

  EAF = 0.82 x 1.14 = 0.934

  Efforts:

  $$E = a \times (KLOC)^b \times EAF = 2.8 \times 400^{1.2} \times 0.934 = 3470PM$$

  Development Time:

  $$D = c \times E^d = 2.5 \times 3470^{0.32} = 33.9M$$

- Case 2

  EAF = 1.29 x 0.95 = 1.22

  Efforts:

  $$E = a \times (KLOC)^b \times EAF = 2.8 \times 400^{1.2} \times 1.22 = 4528PM$$

  Development Time:

  $$D = c \times E^d = 2.5 \times 4528^{0.32} = 36.9M$$

  Since E and D of case one is less, pool one developers are the best option

Detailed Model

The Detailed COCOMO model goes further by partitioning the project into its constituent phases or modules (such as requirements, design, coding, testing, integration). The effort for each phase is estimated separately by applying the basic or intermediate model with additional "phase-sensitive" cost drivers. The overall project effort is the sum of the estimated efforts for each phase.

Development Phases typically include:

- Planning and Requirements
- System Design
- Detailed Design
- Module Code and Test
- Integration and Test
- Cost Construction (final summing)

For each module, the formulas similar to the intermediate model are applied. Then, effort distribution percentages are applied to each phase to determine phase-specific schedules and staffing.

Use of Detailed Estimates

By breaking down the project, the detailed model allows:

- Fine-tuned estimations that account for varying productivity levels in different phases.
- Better insight into staffing and scheduling per phase.
- Improved accuracy when sufficient information is available (this is especially useful for projects with significant complexity or variability among modules).

Numerical 3

Q. Consider a project with the following main components
(1) Screen Edit
(2) CLI
(3) File I/P and O/P
(4) Cursor Movt
(5) Screen Movement

The sizes for these are estimated to be 4k, 2k, 1k, 2k, 3k LOC.
Using COCOMO, determine

I: Overall cost and schedule estimates
(assume values for **cost drivers**, with at least 3 being different from 1.0)

II: Cost and schedule estimates for different phases.

Phase sensitive multiplier for effort

- Planning : 0.06
- System design : 0.16
- Detailed design : 0.26
- Code : 0.42
- Testing : 0.16

Phase sensitive multiplier for duration

- Planning : 0.10
- System design : 0.19
- Detailed design : 0.24
- Code : 0.39
- Testing : 0.18

Solution

Assuming cost drivers:

- RELY= high
- LEXP = low
- CPLX = high
- ACAP = high

EAF = 1.216

Efforts:

$$E = a \times (KLOC)^b \times EAF = 3.2 \times 400^{1.05} \times 1.216 = 52.9PM$$

Development Time:

$$D = c \times E^d = 2.5 \times 52.9^{0.38} = 11.29M$$

solution

- Effort:
  - Plan = 0.06 x 52.9 = 3.174
  - Design = 0.16 x 52.9 = 8.464
  - Detailed design = 0.26 x 52.9 = 13.754
  - Code = 0.42 x 52.9 = 22.218
  - Test = 0.16 x 52.9 =8.464

- Development time:
  - Plan = 0.1 x 11.29 = 1.129
  - Design = 0.19 x 11.29 = 2.1451
  - Detailed design = 0.24 x 11.29 = 2.70
  - Code = 0.39 x 11.29 = 4.4031
  - Test = 0.18 x 11.29 = 2.03

Limitations of COCOMO 1

- **Accuracy:**
  COCOMO 1 provides nominal estimates. Actual costs can vary if, for example, a project is forced to be completed in less time than estimated.
- **Dependency on KLOC:**
  The model is sensitive to the estimated size of the code. Early in a project, estimating KLOC accurately can be challenging.
- **Cost Drivers Subjectivity:**
  In the Intermediate and Detailed models, ratings for cost drivers are subjective and require historical data or expert judgment.
- **Not Agile-Friendly:**
  COCOMO 1 was developed for waterfall-style projects and may be less applicable to iterative or agile environments without modifications.

Comparison with Other Methods

- **Expert Judgment:**
  While expert judgment can be quick, its subjectivity often leads to inconsistent estimates. COCOMO's reliance on historical data and mathematical models reduces bias.
- **Function Point Analysis (FPA):**
  FPA measures functionality delivered rather than lines of code and can be very effective, especially in early design phases. However, FPA requires significant effort to standardize the process and is subject to variability in counting and weighting. COCOMO (especially in its intermediate/detailed forms) complements FPA by translating size estimates into effort and schedule predictions using a well-understood cost model.

Conclusion

The COCOMO 1 model remains a classic tool in software engineering for estimating project effort and schedule. Its three forms—Basic, Intermediate, and Detailed—provide a range of estimation granularity:

- The **Basic Model** is useful for early, rough estimates based solely on size.
- The **Intermediate Model** improves accuracy by incorporating 15 cost drivers.
- The **Detailed Model** divides the project into phases or modules for an even more refined estimate.
- Understanding each project type (Organic, Semi-detached, and Embedded) and applying the proper coefficients are key to using the model effectively. Although modern practices sometimes call for more agile estimation techniques, COCOMO's structured approach and empirical foundation offer a valuable baseline against which many other models are compared.