DUY TAN UNIVERSITY

INTERNATIONAL SCHOOL

SOFTWARE MEASUREMENTS & ANALYSIS

CLASS: CMU-CS 462 HIS

COCOMO CALCULATOR

(Constructive Cost Model Calculator)

MENTOR: **Ph.D Nguyen Duc Man**

TEAM MEMBER: **Bui Duc Huy**

Van Cong Le Ca

Hua Hoang Phuc

Nguyen Thanh Trung

Danang, 23/05/2022

Table Of Contents

1. THEORETICAL BASIS:

1.1. COCOMO:

- Cost estimation: prediction of both the person-effort and elapsed time of a project.
- The Constructive Cost Model (COCOMO) is a procedural software cost estimation model developed by Barry W. Boehm.
- COCOMO predicts the effort and schedule for a software product development based on inputs relating to the **size** of the software and a number of **cost drivers** that affect productivity.
- COCOMO has three different models that reflect the **complexity**:
    - Basic model that is applied early in the project
    - Intermediate model that is applied after requirements acquisition
    - Advanced model that is applied after design is complete
- All three models take the form:

$$E = aS^b \times EAF \qquad T_{dev} = cE^d$$

    - E is effort in person months
    - T dev is the development time
    - S is size measured in KLOC
    - EAF is an effort adjustment factor (1 in the Basic model)
    - Factors a, b, c and d depend on the development mode
- COCOMO Basic Model:
    - Applicable to small to medium sized software projects
    - Use for a quick and rough estimates
    - Three modes of software development are considered:
        - Organic
        - Semi-detached
        - Embedded
    - Organic Mode:
        - A small team of experienced programmers develop software in a very familiar environment
        - Require little Innovation
        - Size range (0-50 KLOC)
    - Semi-detached mode:

- An intermediate mode between the organic mode and embedded mode
- Depending on the problem at hand, the team include the mixture of experienced and less experienced people
- Require medium Innovation
- Development environment is medium
- Size range (50 - 300 KLOC)

- Embedded mode:
  - Project has tight constraints
  - Hard to find experienced persons
  - Require significant Innovation
  - Development environment is complex
  - Size range (over 300 KLOC)

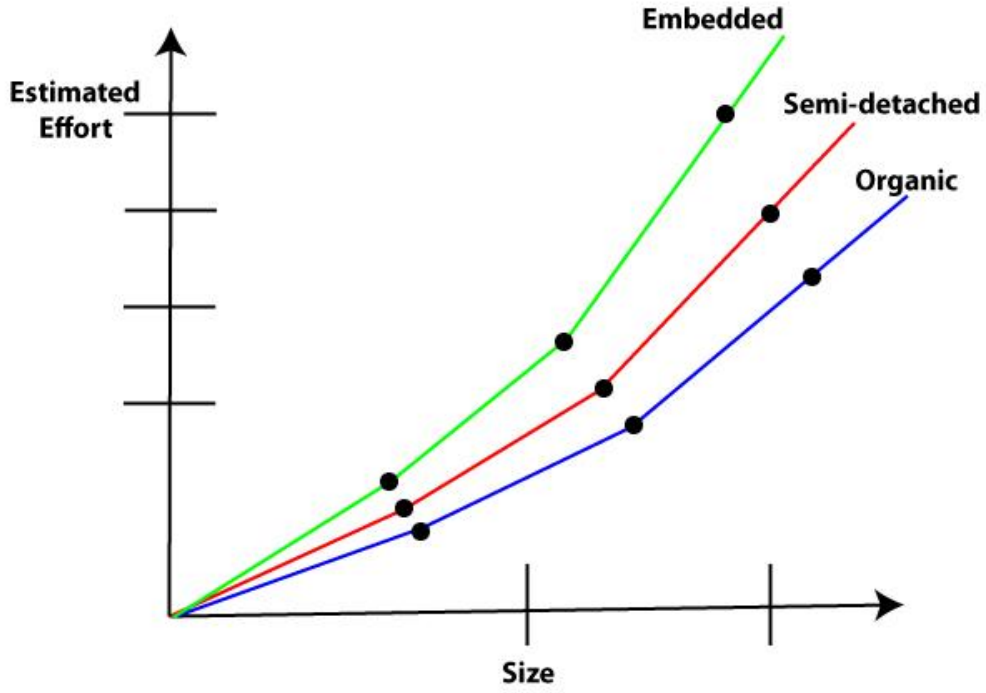
Mode	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

- The Basic COCOMO equation is:

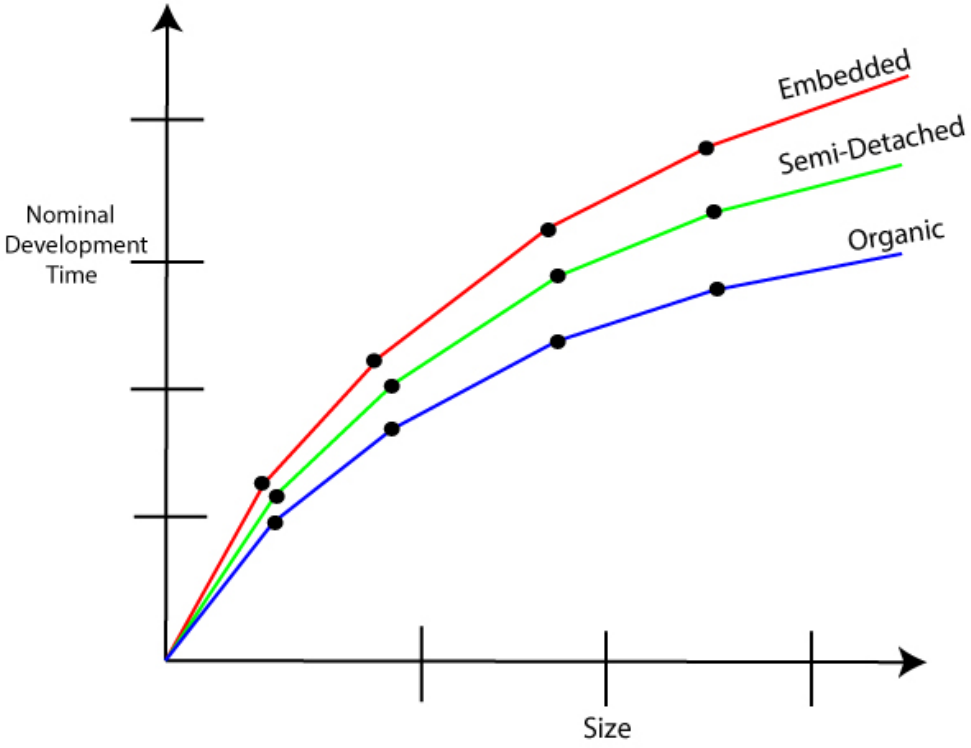
$E_{nom} = a\ (KLOC)^b$ EAF = 1

$T_{dev} = c(E_{nom})^d$

Mode	Effort	Schedule
Organic	$E_{nom} = 2.4*(KLOC)^{1.05}$	$T_{dev} = 2.5*(E_{nom})^{0.38}$
Semi-detached	$E_{nom} = 3.0*(KLOC)^{1.12}$	$T_{dev} = 2.5*(E_{nom})^{0.35}$
Embedded	$E_{nom} = 3.6*(KLOC)^{1.20}$	$T_{dev} = 2.5*(E_{nom})^{0.32}$



Effort versus product size



Development time versus size

- COCOMO Intermediate:
  - Intermediate COCOMO computes software development effort as function of program size and a set of "cost drivers" that include subjective assessment of product, hardware, personnel and project attributes.
  - The intermediate COCOMO equation is:

    $E = a(\text{KLOC})^b \times \text{EAF}$

    $T_{dev} = cE_{nom}^{d}$

    - The factors a, b, c and d for the intermediate model are shown in the table.

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

    - The effort adjustment factor (EAF) is calculated using 15 cost drivers.
  - The cost drivers are grouped into 4 categories: *product*, *platform*, *personnel*, and *project*.
  - Each cost driver is rated on a 6 points ordinal scale ranging from *very low* to *extra high* importance (*very low, low, nominal, high, very high, extra high*). Based on the rating, the *effort multiplier (EM)* is determined (Boehm, 1981). The product of all effort multipliers is the EAF.

$$EAF = \prod_{i=1}^{15} EMi$$

  - Cost Driver Rating:

	Description	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
Platform							
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 machine experience	1.21	1.10	1.00	0.90	-	-
LEXP	Language experience	1.14	1.07	1.00	0.95	-	-
Project							
MODP	Modern programming 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	-

- COCOMO Advanced:
  - Advanced COCOMO model computes effort as a function of program size and a set of cost drivers *weighted according to each phase of 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 [Fenton, 1997].
  - The four phases used in the detailed COCOMO model are:
    - Requirements planning and product design (RPD)
    - Detailed design (DD)
    - Code and unit test (CUT)
    - Integration and test (IT)

1.2. COCOMO II:

- Updated version, called COCOMO II , accounts for recent changes in software engineering technology, including object oriented software, software created via spiral or evolutionary development models, software reuse and building new systems using off the shelf software components.
- COCOMO II includes three stage series of models:
  - The earliest phases will generally involve prototyping, using the Application Composition model capabilities. (replaces Basic model in COCOMO)
  - COCOMO II provides an early estimation model called the Early Design model. (replaces Intermediate model in COCOMO)
  - Once the project is ready to develop and sustain a fielded system, it should have a life cycle architecture, which provides more accurate information on cost driver inputs, and enables more accurate cost estimates. To support this stage, COCOMO II provides the Post Architecture model. (replaces Advanced model in COCOMO)

- Application Composition Model:
    - The Application Composition model is used in early development stages and 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.

$$E = OP / PROD$$

        - OP is the object point
        - PROD is the productivity rate

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

- Early Design model:
    - The Early Design model is used to evaluate alternative software/system architectures and concepts of operation. Unadjusted function point count (UFC) is used for sizing. This value is converted to KLOC.
    - The Early Design model's equation is:

$$E = 2.45 \times KLOC \times EAF$$

    - The effort adjustment factor (EAF) is calculated as in the original COCOMO model using 7 cost drivers:
        - Product reliability and complexity (RCPX)
        - Required reuse (RUSE)
        - Platform difficulty (PDIF)
        - Personnel capability (PERS)
        - Personnel experience (PREX)
        - Facilities (FCIL)
        - Schedule (SCED)
- Post Architecture model:

- The Post-Architecture model involves the actual development and maintenance of a software product using source instructions and / or function points for sizing, with modifiers for reuse and software breakage; a set of 17 cost drivers (EAF) and a set of 5 scale factors (SF) determining the projects scaling component.
- The Post Architecture model equation is:

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

$$b = 0.91 + 0.01\sum_{i=1}^{5} SFi$$

$$EAF = \prod_{i=1}^{15} EMi$$

$$T_{dev} = c(E)^{(0.28 + 0.2(b - 0.91))} \times SCED\%100$$

  - a is a constant derived from historical project data ($a=2.94$ in COCOMO II 2000)
  - SF_i is a weighting factor for the i^{th} scale driver
  - EM_i is the effort multiplier for the i^{th} cost driver
  - SCED is the compression/expansion percentage in the SCED cost driver

	Very Low	Low	Nominal	High	Very High	Extra High
SCED	75% of Nominal	85%	100%	130%	160%	

2. TOOL:

2.1. UI:

2.2. ALGORITHM:

3. TEST CASE:

3.1. COCOMO:

3.2. COCOMO II:

4. CONCLUDING:

COCOMO Uses for Software Decision Making:

Making investment decisions and business-case analyses

Setting project budgets and schedules

Performing tradeoff analyses

Cost risk management

Development vs. reuse decisions

Legacy software phaseout decisions

Software reuse and product line decisions

Process improvement decisions

5. REFERENCES:

Boehm, Barry (1981). Software Engineering Economics. Prentice-Hall.

Barry Boehm, Chris Abts, A. Winsor Brown, Sunita Chulani, Bradford K. Clark, Ellis Horowitz, Ray Madachy, Donald J. Reifer, and Bert Steece. Software Cost Estimation with COCOMO II (with CD-ROM). Englewood Cliffs, NJ:Prentice-Hall, 2000.

COCOMO (Constructive Cost Model), Seminar on Software Cost Estimation WS 2002 / 2003 presented by Nancy Merlo – Schett, Prof. Dr. Martin Glinz & Arun Mukhija.