



# COCOMO

## Constructive Cost Model

# Estimasi biaya dan waktu (1)

- ◆ **Top down (analogi histori dan informasi):** dari analisa bisnis sampai ke detail.
- ◆ **Bottom up:** dari estimasi masing-masing aktivitas proyek dikumpulkan secara total.
- ◆ **Model matematis;**
- ◆ **Software tools.**
- ◆ Perlu diingat dalam SW metodologi bahwa:
  - Biaya (cost) tidak sebanding linear dengan jumlah code yang akan diprogram (size).

# Estimasi biaya dan waktu (2) Model matematis

- ◆ Dasar perhitungan:

$$\text{effort} = C \times \text{size}^M$$

- ◆ Dikenal sebagai Constructive Cost Model (COCOMO), model konstruksi biaya.
- ◆ C dan M adalah koefisien konstanta ( $> 1$ ), tergantung pada *tipe proyek* dan *organisasi*, dengan cara melihat Tabel Konstanta (sudah tersedia dari penelitian).
- ◆ Ditentukan pula oleh: *application experience, leadership capability, new environment and tools, requirements uncertainty, software reuse.*

# COCOMO Model

## ◆ **Basic** (COCOMO I 1981)

Menghitung dari estimasi jumlah FP dan LOC;

FP = suatu unit pengukuran untuk keterhubungan dan keterkaitan antar prosedur, fungsi dan lingkungan SW

## ◆ **Intermediate** (COCOMO II 1999)

- Menghitung dari besarnya program dan "*cost drivers*" (faktor-faktor yang berpengaruh langsung kepada proyek), spt: hardware, personnel, dan atribut-atribut proyek;

## ◆ **Advanced**

- Memperhitungkan semua karakteristik dari "*intermediate*" di atas dan "*cost drivers*" dari setiap fase (analysis, design, implementation, etc) dlm *SW life cycles*;

# Basic COCOMO (1)

(**E** = *effort*) = **Ca x (size=KLOC=kilo line of code) <sup>Ma</sup>**  
(satuan: **ManMonth** (Person Month) = 152 jam kerja)

(**D** = *duration*) = **Cb x E <sup>Mb</sup>**  
(satuan: **Month**)

**Productivity** = **size / E** (satuan: **KLOC/Man Month**)

**Average staffing** = **E / D** (satuan: **FTE** = Full Time Employees → jumlah orang yang bekerja penuh dalam 1 hari kerja ~ 8 jam )

# Basic COCOMO (2)

- 1: Menghitung estimasi informasi **nilai domain** → *count total*;
- 2: Menyesuaikan kompleksitas proyek berdasarkan faktor pemberat dan “cost drivers” kemudian menghitung estimasi jumlah **Function Points** → *unit of measure that represent functions required by the user.*

$$\mathbf{FP = count\ total * [0.65 + 0.01 * \sum Fi];}$$

- 3: Menghitung estimasi **LOC** (Line of Code). Tekniknya sama dengan PERT Calculation (three points estimation);

$$\mathbf{EV = (Sopt + 4 Sm + Spess) / 6;}$$

Atau menghitung **LOC / FP dari tabel** berdasar pada bahasa pemrograman;

- 4: Memilih kompleksitas proyek (menentukan **C** dan **M**), dari organic, embedded atau semi-detached system mode.
- 5: Menghitung **E** dan **D** → estimasi biaya dan waktu.

# Basic COCOMO (3)

## ◆ Informasi nilai domain

	(Simple	Avrg.	Compl.)	Jumlah	
Jumlah input pemakai	3	4	6	*	=
Jumlah output pemakai	4	5	7	*	=
Jumlah inquiry pemakai	3	4	6	*	=
Jumlah file	7	10	15	*	=
Jumlah eksternal interface	5	7	10	*	=
					+ <hr/>
					<b><i>count total</i></b>

# Basic COCOMO (4)

- ◆ **Input pemakai:** setiap input data dari user yang dipakai untuk menjalankan aplikasi.
- ◆ **Output pemakai:** setiap hasil output dari proses yang ditampilkan kepada user.
- ◆ **Inquiry pemakai:** setiap *on-line* input yang menghasilkan responsi software secara langsung.
- ◆ **Jumlah file:** setiap master file yang menjadi bagian dari aplikasi.
- ◆ **Eksternal interface:** setiap interface (sarana) eksternal yang menyalurkan informasi dari sistem satu ke sistem lainnya.



# Basic COCOMO (5)

- ◆ **Ada 14 pos kompleksitas faktor (cost drivers), yaitu:**
  1. *Backup dan recovery*
  2. Komunikasi data
  3. Proses terdistribusi
  4. Kepentingan performa
  5. Keberadaan lingkungan operasi
  6. *Online data entry*
  7. Input melalui bbrp tampilan/operasi
  8. Peng-update-an file master secara online
  9. Kompleksitas nilai 'domain' (tahap1) diatas
  10. Kompleksitas proses internal aplikasi
  11. Perulangan (*reuse*) penggunaan code
  12. Ketersediaan rancangan untuk konversi dan instalasi
  13. Rancangan untuk pengulangan instalasi di lingkungan yg berbeda
  14. Fleksibilitas bagi pemakai
- ◆ **Kesemuanya ini dihitung berdasarkan nilai dari 0-5 menunjukkan perkiraan nilai kepentingan**  
**(*No Influence, Incidental, Moderate, Average, Significant, Essential*)**

# Tabel LOC / FP Rata-rata

Programming Language	LOC / FP (rata-rata)
Bahasa Assembly	320
C	128
COBOL	105
Fortran	105
Pascal	90
Ada	70
Bahasa Berorientasi Obyek	30
Bahasa Generasi Keempat (4GLs), yaitu bahasa yang digunakan spesifik untuk suatu tools, biasa untuk aplikasi database, contoh: PL/SQL dalam Oracle.	20
Generator Kode	15
Spreadsheets	6
Desain Grafis (icons)	4

## Basic COCOMO (5)

<b>Software Project</b>	<b><i>Ca</i></b>	<b><i>Ma</i></b>	<b><i>Cb</i></b>	<b><i>Mb</i></b>
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

- ◆ **Organic** = kecil, sederhana (co, pembuatan situs mandiri untuk perusahaan);
- ◆ **Semi-detached** = menengah (co. transaksi sistem pada database sebuah bank);
- ◆ **Embedded** = kompleksitas tinggi, ketergantungan pada lingkungan aplikasi lainnya (co. aplikasi pengontrolan pada pesawat terbang).

# Latihan: Basic COCOMO

- ◆ Dalam soal ini Anda akan mensimulasikan perhitungan estimasi jumlah kode pada suatu proyek perangkat lunak dengan menggunakan COCOMO model. Seluruh data-data yang diperlukan akan diberikan, tugas anda hanyalah menerapkan proses perhitungannya. Latar belakang proyek adalah:
- ◆ **Suatu perangkat lunak CAD akan menerima data geometri dua dan tiga dimensi dari seorang perekayasa. Perekayasa akan berinteraksi dan mengontrol sistem CAD melalui suatu interface pemakai yang akan memperhatikan desain manusia mesin yang baik. Semua data geometri dan informasi pendukung yang lain akan dipelihara pada database CAD. Modul analisis desain akan dikembangkan untuk memproduksi output yang dibutuhkan yang akan ditampilkan pada berbagai perangkat grafik. Perangkat lunak akan dirancang untuk mengontrol dan berinteraksi dengan perangkat keras peripheral termasuk mouse, digitizer dan printer laser.**  
Dengan latar belakang tersebut, dibuatlah sebuah desain perangkat lunak dengan data flow diagram, yang pada akhirnya dapat digunakan untuk mengestimasi jumlah kode yang diperlukan dalam pengimplementasian dengan memperkirakan jumlah *Function Point*-nya.

# Tahap 1: Informasi Nilai Domain

<i>Informasi nilai domain</i>	<i>Jumlah estimasi per domain</i>	<i>Bobot domain <u>Average</u></i>	<i>Jumlah FP per domain ( ?? )</i>
Jumlah input	24	4	
Jumlah output	26	5	
Jumlah inquiry	22	4	
Jumlah file	4	10	
Jumlah interface eksternal	2	7	
Jumlah total			

# Tahap 2: Hitung Cost Drivers / Kompleksitas

## **FAKTOR**

## **Bobot kepentingan**

1. Backup dan recovery	4
2. Komunikasi data	2
3. Proses terdistribusi	0
4. Kepentingan performa	4
5. Keberadaan lingkungan operasi	3
6. Entri data online	4
7. Input melalui beberapa tampilan / operasi	5
8. Peng- <i>update</i> -an file master secara online	3
9. Kompleksitas nilai 'domain' (tahap1) diatas	5
10. Kompleksitas proses internal aplikasi	5
11. Perulangan ( <i>reuse</i> ) penggunaan code	4
12. Ketersediaan rancangan untuk konversi dan instalasi	3
13. Rancangan untuk pengulangan instalasi di lingkungan yang berbeda	5
14. Fleksibilitas bagi pemakai	5

Hitung **total kompleksitas** yang ada  $\Sigma F_{ij} = \dots$

Hitung estimasi FP-nya **FP = jumlah total nilai domain \* (0.65 + {0.01 \*  $\Sigma F_{ij}$ })**

# Tahap 3: Hitung Size

◆ Diminta bahwa Anda menggunakan bahasa pemrograman yang berorientasi obyek. Diketahui bahwa estimasi **LOC / FP**-nya adalah 30.

◆ Hitung estimasi LOC-nya:

LOC = ... *ubah menjadi KLOC*

KLOC = ...

# Tahap 4: Konstanta Kompleksitas Proyek

Type sistem	$ca$	$Ma$	$cb$	$Mb$
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

- ◆ Dari hasil requirements dan desain, diketahui bahwa produk akan beroperasi di tipe sistem ***semi-detached***.
- ◆ Tentukan nilai konstanta, dari tabel tipe-sistem  
 $Ca = \dots$                        $Cb = \dots$   
 $Ma = \dots$                        $Mb = \dots$



# Tahap 5: Hitung Besaran COCOMO

Effort = ...

Durasi = ...

Produktivitas = ...

Staffing = ...