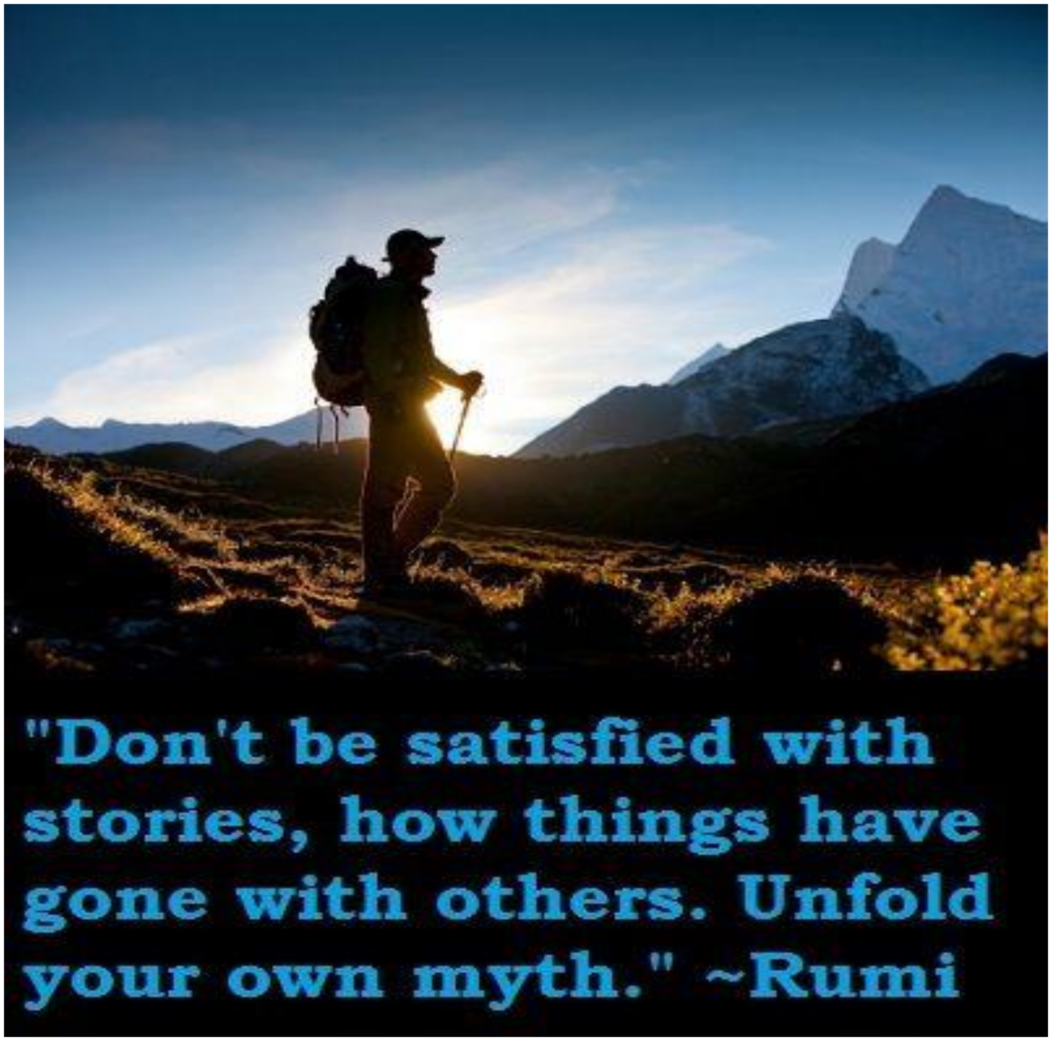
بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

In the name of Allah the most Beneficial ever merciful



# *Artificial Intelligence (AI) in Software Engineering*

## Effort Estimation



*Department of Computer Science , Univeristy of Karachi (DCS-UBIT)*
*4th April 2020*

# Class Quiz 6th April 2021   9:30 am

Must bring empty printout of  Effort estimation  template in  separate clip file for Quiz and Submission.

# Agenda

0- Concepts Recap ( AI, Linear Regression, ML, Model)

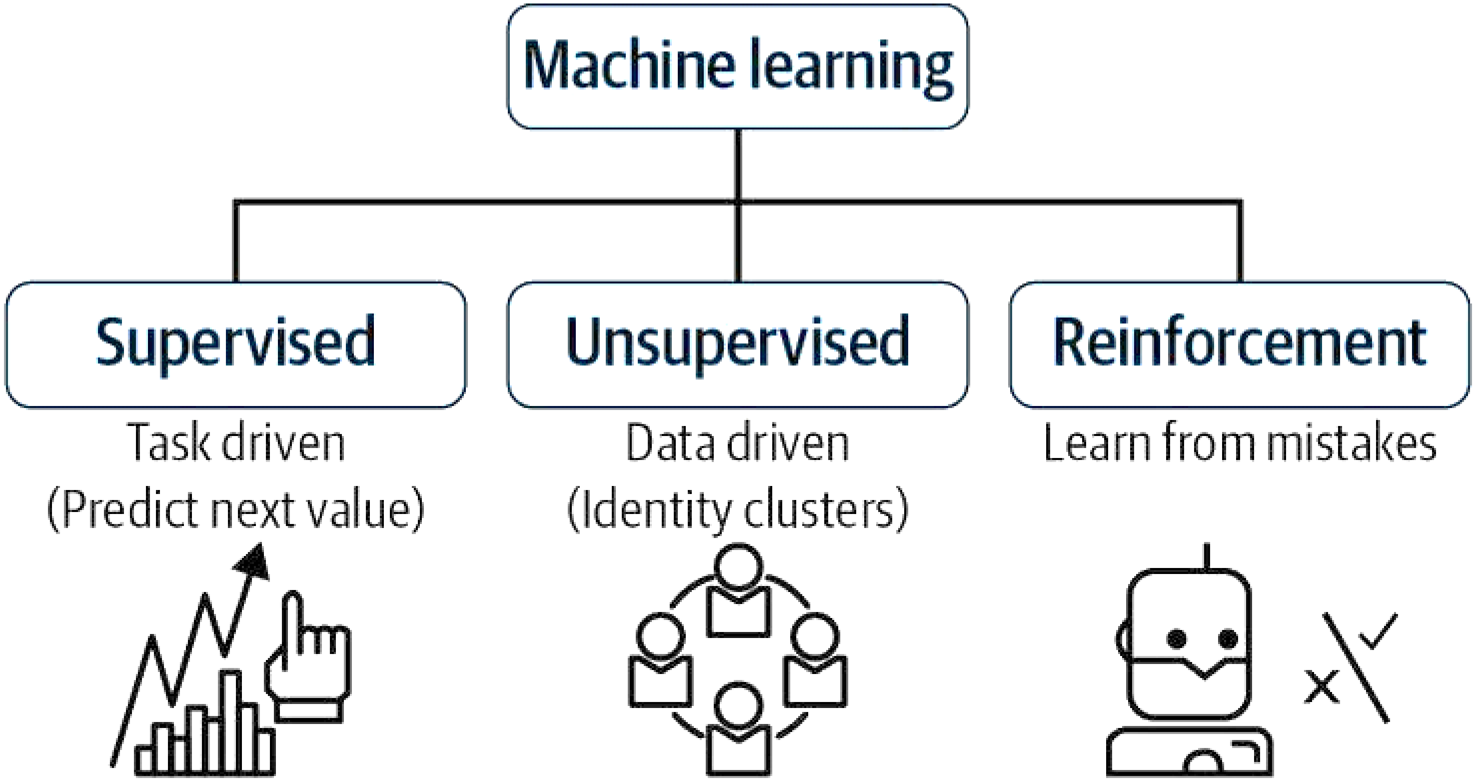
1- Basic CoCoMo

2- Intermediat CoCoMo

3- Functional Points (FP)


QUIZ


SUCCESS


Machine learning
Supervised
Unsupervised
Reinforcement
Task driven
(Predict next value)
Data driven
(Identity clusters)
Learn from mistakes

# First Moment and SSE

| Calculation of the mean of a "sample of 100" | | |
|---|---|---|
| Column A<br>Value or Score<br>(X) | Column B<br>Deviation Score ($\Delta$) (X-Xbar) | Column C<br>Deviation Score² ($\Delta^2$) (X-Xbar)² |
| 100 | 100-94.3 = 5.7 | $(5.7)^2 = 32.49$ |
| 100 | 100-94.3 = 5.7 | $(5.7)^2 = 32.49$ |
| 102 | 102-94.3 = 7.7 | $(7.7)^2 = 59.29$ |
| 98 | 98-94.3 = 3.7 | $(3.7)^2 = 13.69$ |
| 77 | 77-94.3 = -17.3 | $(-17.3)^2 = 299.29$ |
| 99 | 99-94.3 = 4.7 | $(4.7)^2 = 22.09$ |
| 70 | 70-94.3 = -24.3 | $(-24.3)^2 = 590.49$ |
| 105 | 105-94.3 = 10.7 | $(10.7)^2 = 114.49$ |
| 98 | 98-94.3 = 3.7 | $(3.7)^2 = 3.69$ |
| $\sum X$ | $\sum\Delta$ or $\sum$(X-Xbar) | $\sum\Delta^2$ or $\sum$(X-Xbar)² |
| | "first moment" | Sum of Squares (SS) |

UNIVERSITY OF KARACHI

# Variance and SD

**Variance.** ***The sum of squares gives rise to variance.*** The first use of the term SS is to determine the variance. Variance for this sample is calculated by taking the sum of squared differences from the mean and dividing by N-1:

$$\text{Variance} = s^2 = \frac{SS}{N-1} = \frac{\Sigma(x-\bar{x})^2}{N-1}$$

**Standard deviation.** ***The variance gives rise to standard deviation.*** The second use of the SS is to determine the standard deviation. Laboratorians tend to calculate the SD from a memorized formula, without making much note of the terms.

$$SD = \sqrt{s^2} = \sqrt{\frac{SS}{N-1}} = \sqrt{\frac{\Sigma(x-\bar{x})^2}{N-1}}$$

UNIVERSITY OF KARACHI

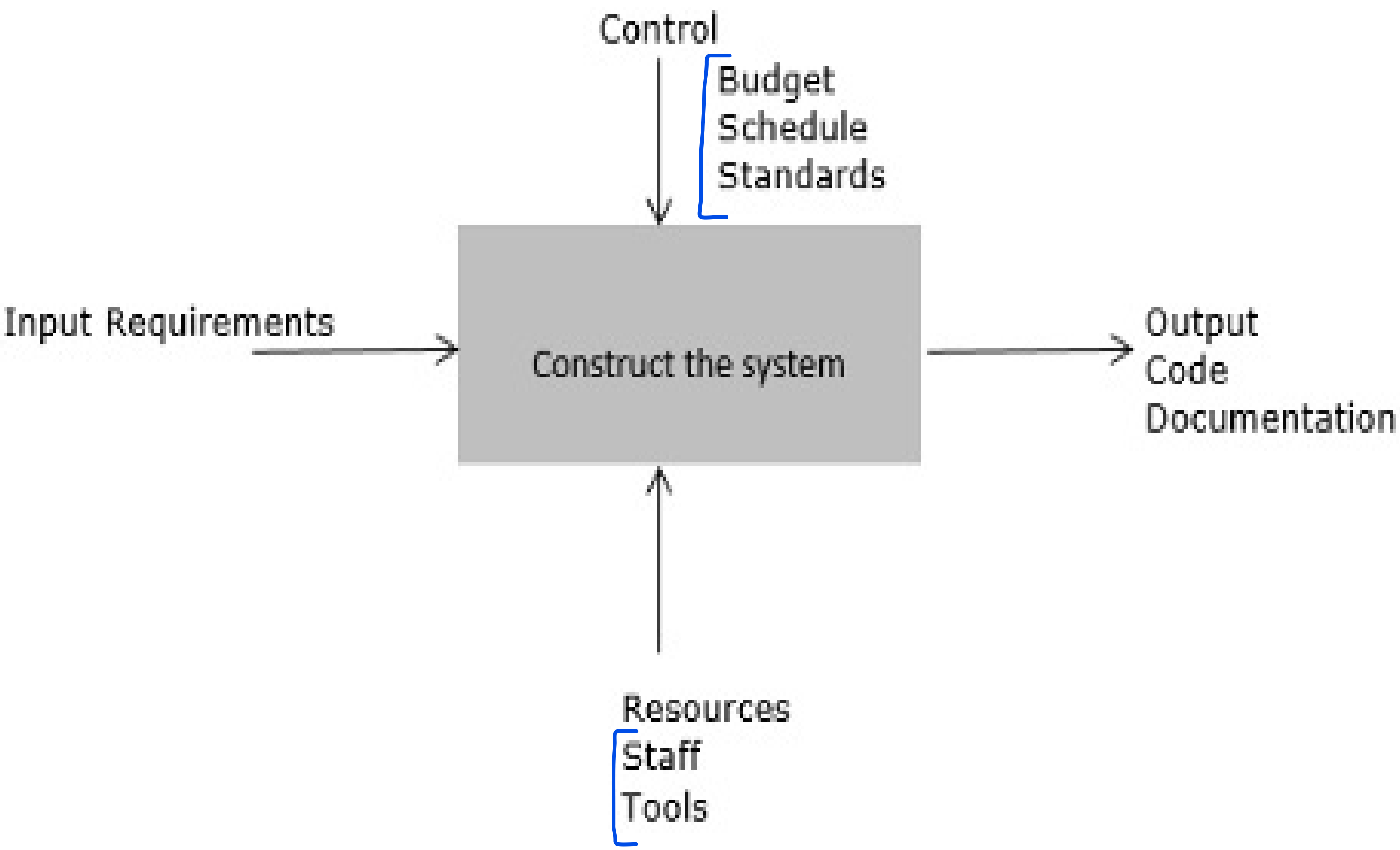
# Variance leads to Project management

## Why Estimate ??

| | Project management activities | |
|---|---|---|
| During project start-up | • Estimation<br>• Staffing<br>• Resource acquisition<br>• Training | • Developing the work plan: activities, schedule, resources, and budget allocation |
| During project execution | • Quality assurance control<br>• Reporting and tracking<br>• Metrics collection<br>• Risk monitoring and mitigation<br>• Configuration management<br>• Process Improvement | • Budget control<br>• Schedule control<br>• Requirements control<br>• Verification and validation<br>• Documentation<br>• Problem resolution<br>• Subcontractor management |
| During project closeout | • Product acceptance<br>• Staff reassignment<br>• User training<br>• Product installation | • Archiving<br>• Post-mortem evaluation and assessment<br>• Integration and conversion |

UNIVERSITY OF KARACHI

# Why Estimate ??



# Why Estimate ??

## Specification for Development Plan

- Project
- Feature List
- Development Process
- Size Estimates
- Staff Estimates
- Schedule Estimates
- Organization
- Gantt Chart

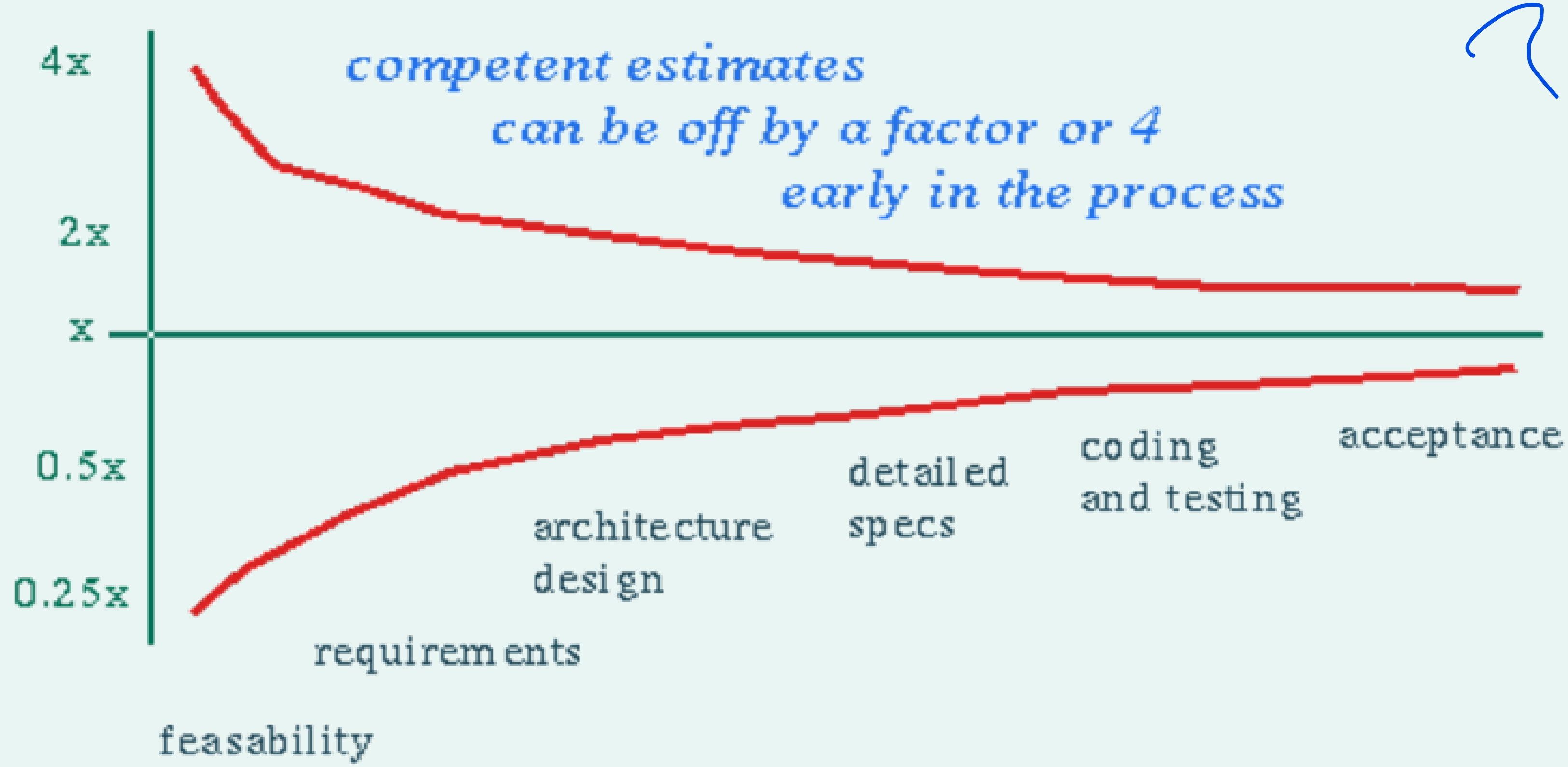
5

**(A) Identify widest variance from graph ?**

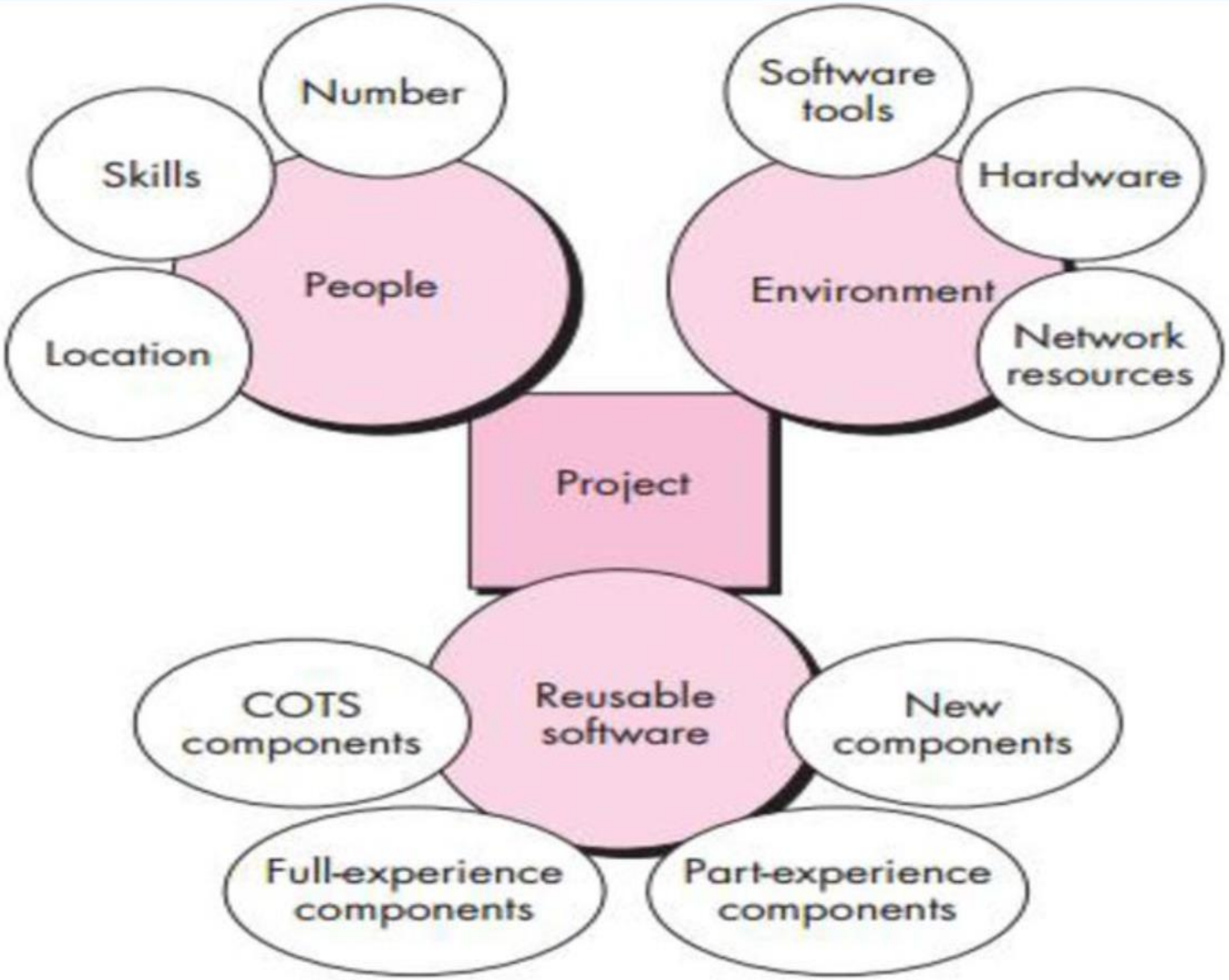
**(B) What are four types of feasibility ?**

*Rule of the World*

The worst estimating performance occurrs just where we need the best.



# Type and attributes of Resources




UNIVERSITY OF KARACHI

# Models -Equations-Hypothesis


Name equations/models that are used to fit the data ??

What is mean by hypothesis in this context ??

What data and attributes we want to talk about??

[COCOMO Resource Estimation Model (unc.edu)]

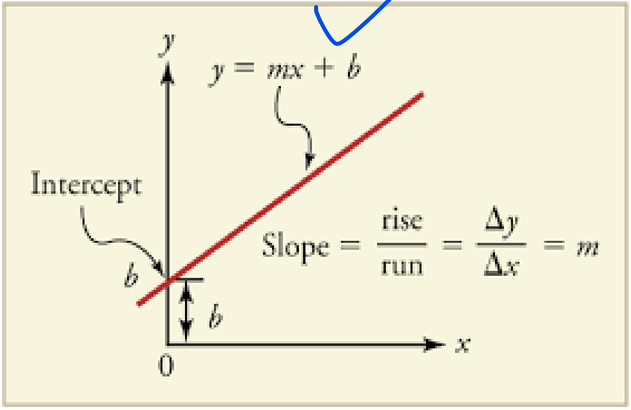
UNIVERSITY OF KARACHI

# Small Projects

The magnitude of the effort is a linear function of the size of the project.

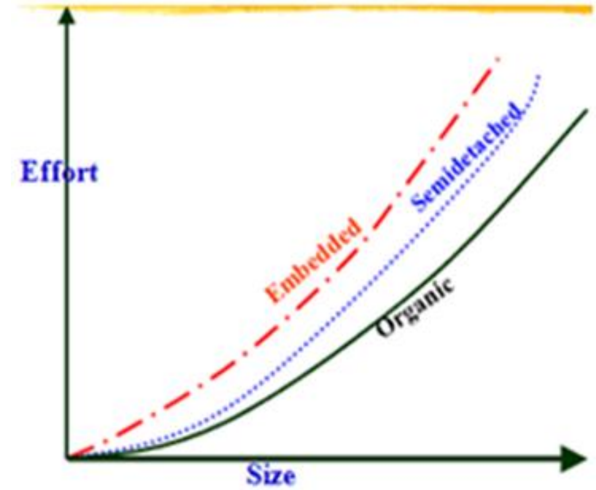
Linear model holds up until a certain point with a team of 2-3 people.

$$Effort = a * size + b$$



# Larger Projects

The effort behaviour of people is non-linear for Larger projects with team > 3 people.




$$Effort = a * size^b$$

Based on complexity of project

Usually LOC but may also be FP (functional points

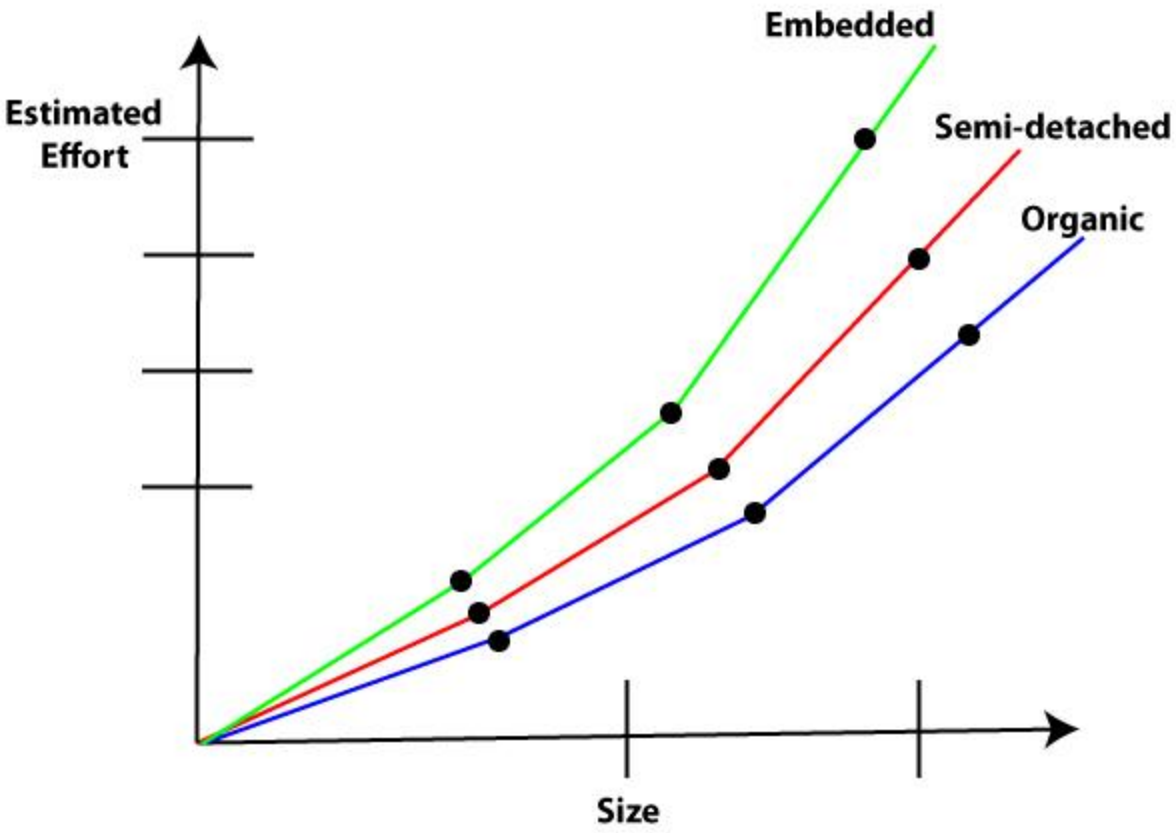
Empirically derived

UNIVERSITY OF KARACHI

# Effort vs. Product size

Effort required to develop a product increases very rapidly with project size.

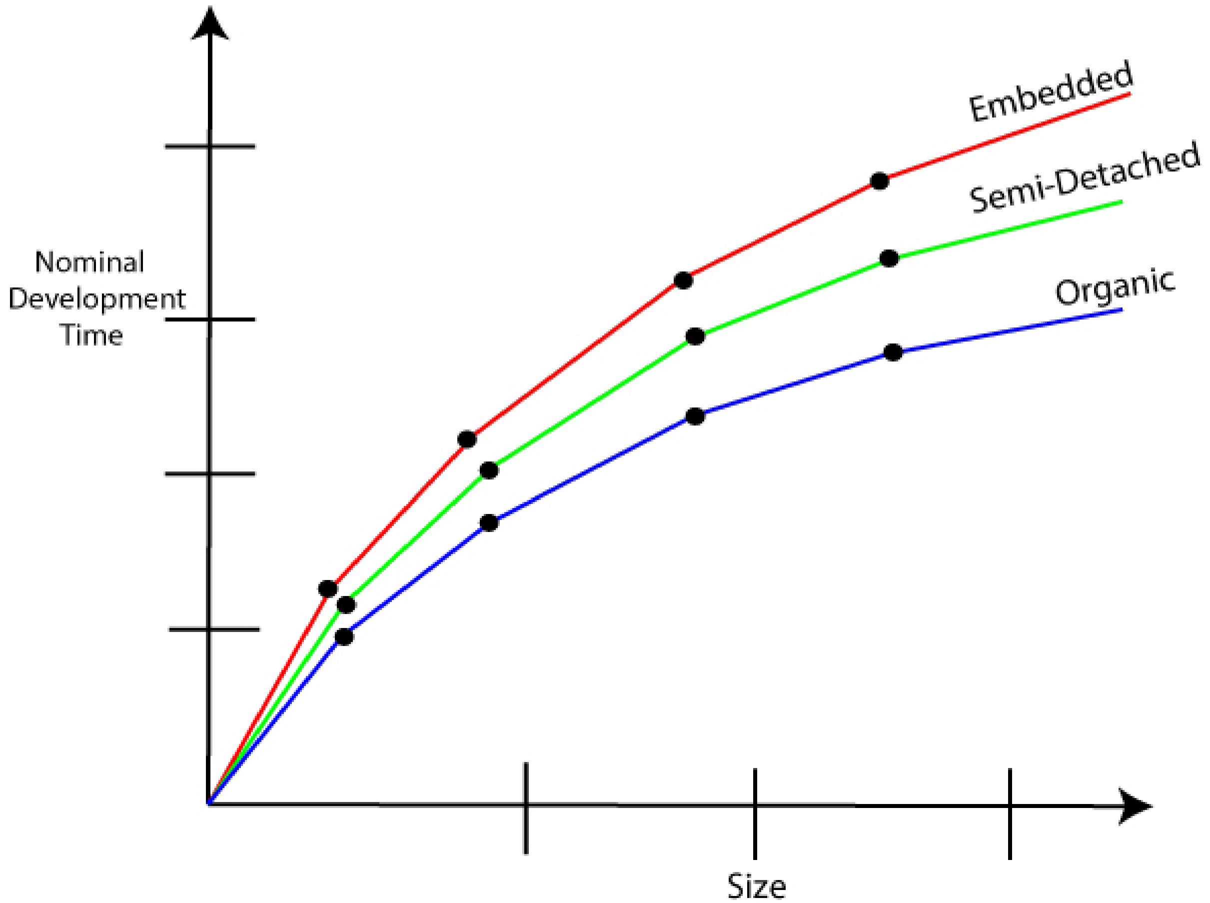
?




Effort versus product size

# Time vs. Product Size

Development time is roughly the same for all three categories of products




Development time versus size

# Size Oriented Metrics

- ✓ Errors per KLOC (thousand lines of code) • Defects per KLOC
- ✓ $ per KLOC
- ✓ Pages of documentation per KLOC
- ✓ Errors per person-month
- ✓ KLOC per person-month
- ✓ $ per page of documentation

LOC ≡ Line of Code

KLOC ≡ Thousands of LOC

KSLOC ≡ Thousands of Source LOC

NCSLOC ≡ New or Changed KSLOC

| Project | LOC | Effort | $(000) | Pp. doc. | Errors | Defects | People |
|---|---|---|---|---|---|---|---|
| alpha | 12,100 | 24 | 168 | 365 | 134 | 29 | 3 |
| beta | 27,200 | 62 | 440 | 1224 | 321 | 86 | 5 |
| gamma | 20,200 | 43 | 314 | 1050 | 256 | 64 | 6 |
| • | • | • | • | • | • | | |
| • | • | • | • | • | • | | |
| • | • | • | • | • | • | | |

# Early Stage Effort estimation

| Method | Type | Strengths | Weaknesses |
|---|---|---|---|
| 1 COCOMO Model | Algorithmic | Universal Approach; More predictable and accurate | Much historical data is required; V |
| 2 Function Point FP | Algorithmic | Language independent | Quite time consuming; Complex to use |

# Popular Estimation Methods

Linear Programming as a Baseline for Software Effort Estimation Federica Sarro, University College London Alessio Petrozziello, University of Portsmouth

Parametric Estimation

Wideband Delphi

Cocomo

SLIM (Software Lifecycle Management)

SEER-SEM

Function Point Analysis

PROBE (Proxy bases estimation, SEI CMM)

Planning Game (XP) Explore-→Commit

Program Evaluation and Review Technique (PERT)

# Estimation Techniques

- ✓ Task breakdown and effort estimates
- ✓ Size (e.g., Function Points) estimates

Traditional techniques

- ✓ Process based Estimation
- ✓ Estimation with Use-Cases
- ✓ Empirical Estimation Model e.g., COCOMO
- ✓ COCOMO II
- ✓ Linear Programming
- ✓ Machine Learning

# Basic

# CoCoMo

# Boehm's 1981 Postulates

**Organic Mode:**

- ✓ Relatively simple & small projects
- ✓ Small team can handle
- ✓ Good application experience to less rigid requirements
- ✓ requires little innovation means well understood application/program

**Semidetached Mode:**

- ✓ Little Complex compared to organic mode projects in terms of size
- ✓ Team with mixed experience level is required handle
- ✓ rigid requirements
- ✓ less rigid requirements

**Semidetached Mode:**

- ✓ Complex project
- ✓ Tight set of hardware
- ✓ software operational constraints
- ✓

UNIVERSITY OF KARACHI

# Basic CoCoMo Development Modes

| Mode | Project Size | Nature of Project | Innovation | Deadline |
|---|---|---|---|---|
| Organic | Typically 2-50 KLOC | Small size project, Experienced developers. | Little | Not Tight |
| Semi Detached | Typically 50-300KLOC | Medium size project and team. | Medium | Medium |
| Embedded | Typically over 300KLOC | Large project, Real-time systems | Significant | Tight |

costtools.pm (csiac.org)

UNIVERSITY OF KARACHI

# Project characteristics

| | Organic | Semi-detached | Embedded |
|---|---|---|---|
| Project size (lines of source code) | 2,000 to 50,000 | 50,000 to 300,000 | 300,000 and above |
| Team Size | Small | Medium | Large |
| Developer Experience | Experienced developers needed | Mix of Newbie and experienced developers | Good experience developers |
| Environment | - Familiar Environment | - Less familiar environment | - Unfamiliar environment (new)<br>- Coupled with complex hardware |
| Innovation | Minor | Medium | Major |
| Deadline | Not tight | Medium | Very tight |
| Example(s) | Simple Inventory Management system | New Operating system | Air traffic control system |

UNIVERSITY OF KARACHI

# Basic CoCoMo Input and output

Input: Programm Size

CoCoMo model
(regression model)

estimated thousands of lines of code(KLoC)

Effort [man month] $E = a_b (\mathrm{KLoC})^{b_b}$

Development time [month] $T = c_b (\mathrm{E})^{d_b}$

Persons required: $P = E/\mathrm{T}$

Used coefficients

| Project | $a_b$ | $b_b$ | $c_b$ | $d_b$ |
|---|---|---|---|---|
| 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 |

Output:
Calculation of costs using pricing per hour

UNIVERSITY OF KARACHI

# Tuesday 6th April 2021

QUIZ

ORGANIC

a b

$$\text{Person Months} = 2.4 * \text{KDSI}^{1.05}$$

SEMI-DETACHED

$$\text{Person Months} = 3.0 * \text{KDSI}^{1.12}$$

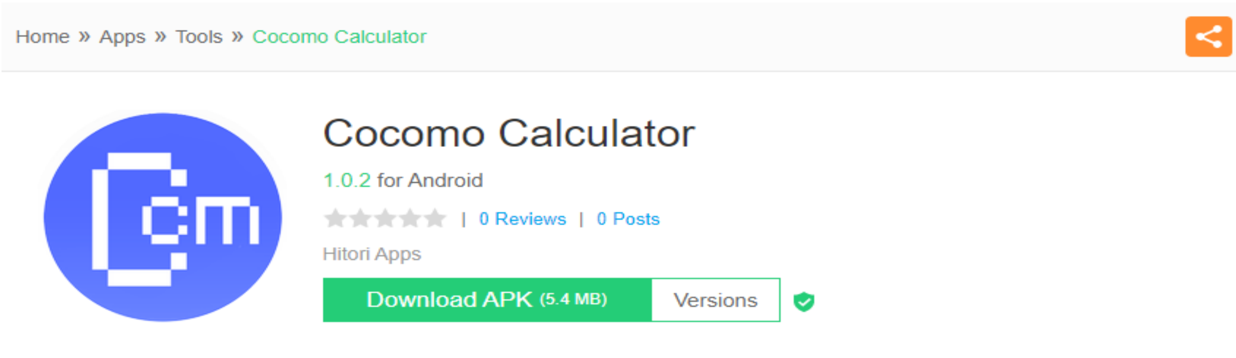
EMBEDDED

$$\text{Person Months} = 3.6 * \text{KDSI}^{1.20}$$

This will normally be represented as a table, which presumes we know the basic form of the model:

| Basic COCOMO | a | b |
|---|---|---|
| ORGANIC | 2.4 | 1.05 |
| SEMI-DETACHED | 3.0 | 1.12 |
| EMBEDDED | 3.6 | 1.20 |

# CoCoMo Calculator User Interface (App)




[Mike's Basic COCOMO calculator (umich.edu)]

[COCOMO calculation (umich.edu)]

# Tuesday 6th April 2021

QUIZ

Fill in the table with examples of your own.

Similarity Rate ~ zero

| Project Title | Type (O/D,E) | Justification |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

UNIVERSITY OF KARACHI

# Tuesday 6th April 2021


**Effort(E) =** $a_b * (KLOC)^{b_b}$ (in Person-months)

**DevelopmentTime(D) =** $c_b * (E)^{d_b}$ (in month)

**Average staff size(SS) = E/D** (in Person)

**Productivity(P) = KLOC / E** (in KLOC/Person-month)

UNIVERSITY OF KARACHI

# Intermediate

# CoCoMo

[COCOMO II - Constructive Cost Model (softwarecost.org)]

# Intermediate CoCoMo

Intermediate CoCoMo rated project characteristics on a scale of 1 to 5

$$Effort = EAF * a * size^{b}$$

Each project characteristic/attribute gives an adjustment factor (from the table) and all factors are multiplied together to get total EAF.

# Boehm Introduced 15 Cost Drivers

Extension of Basic COCOMO

Why Use ?

Basic model lacks accuracy

Computes software development effort as a function of program size and set of 15 Cost Drivers

**Cost Driver**: A multiplicative factor that determines the effort required to complete the software project.

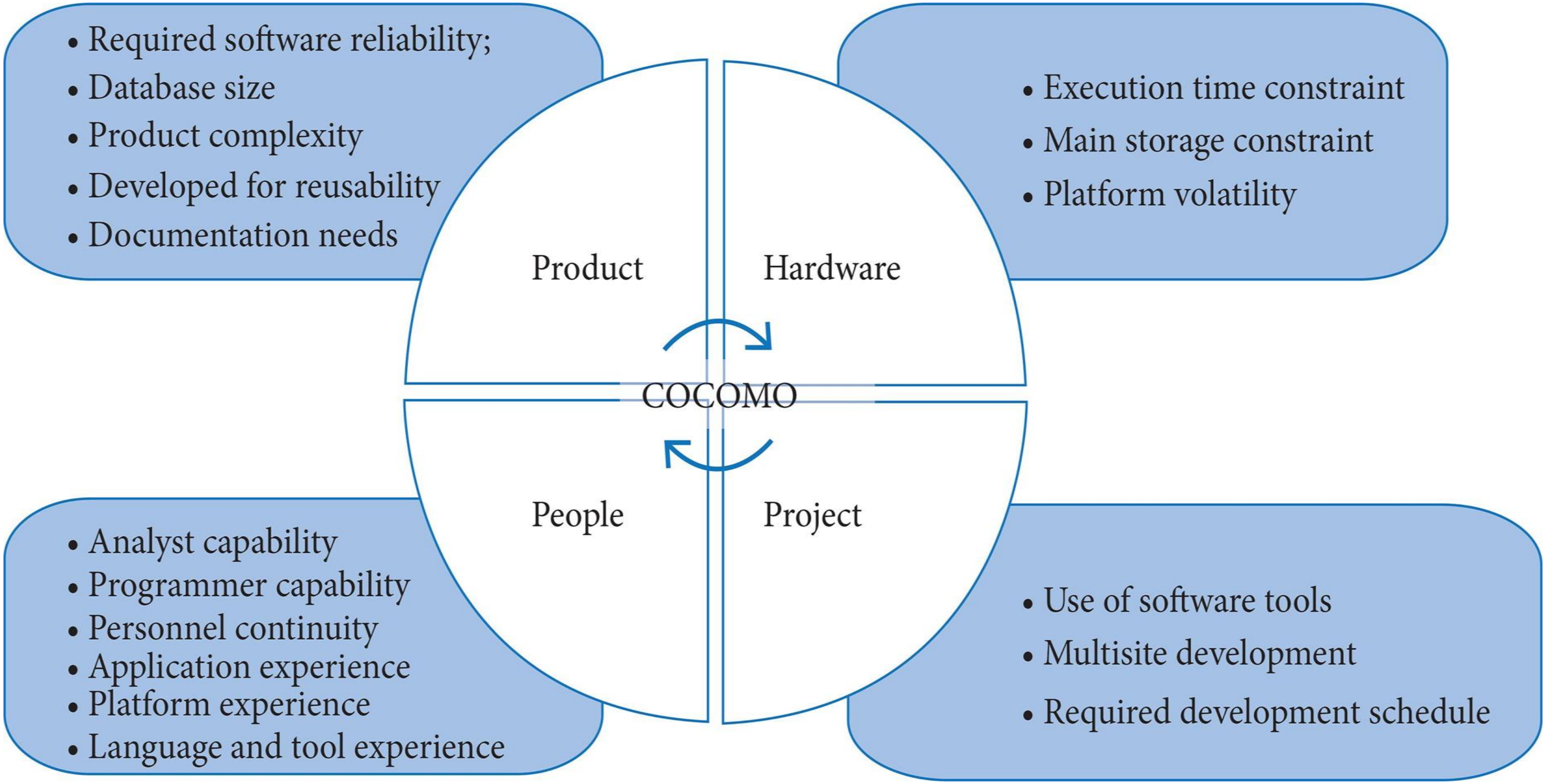
**Why Cost Drivers**?

Adjust the nominal cost of a project to the actual project Environment.

For each Characteristics, Estimator decides the scale factor

Very Low | Low | Nominal | High | Very High | Extra High

# Estimator → Project Manager




UNIVERSITY OF KARACHI

# Effort Adjustment Factor (EAF)

## Project Characteristics Table

Cost adjustments for computing the EAF (Effort Adjustment Factor)

| | v.low | low | nominal | high | v.high | ex.high |
|---|---|---|---|---|---|---|
| **product attributes** | | | | | | |
| required software reliability | 0.75 | 0.88 | 1.00 | 1.15 | 1.40 | |
| database size | | 0.94 | 1.00 | 1.08 | 1.16 | |
| product complexity | 0.70 | 0.85 | 1.00 | 1.15 | 1.30 | 1.65 |
| **computer attributes** | | | | | | |
| execution time constraints | | | 1.00 | 1.11 | 1.30 | 1.66 |
| main storage constraints | | | 1.00 | 1.06 | 1.21 | 1.56 |
| virtual machine volitility | 0.87 | 1.00 | 1.15 | 1.30 | | |
| computer turnaround time | | 0.87 | 1.00 | 1.07 | 1.15 | |
| **personnel attributes** | | | | | | |
| analyst capability | 1.46 | 1.19 | 1.00 | 0.86 | 0.71 | |
| applications experience | 1.29 | 1.13 | 1.00 | 0.91 | 0.82 | |
| programmer capability | 1.42 | 1.17 | 1.00 | 0.86 | 0.70 | |
| virtual machine experience | 1.21 | 1.10 | 1.00 | 0.90 | | |
| programming language experience | 1.14 | 1.07 | 1.00 | 0.95 | | |
| **project attributes** | | | | | | |
| use of modern programming practices | 1.24 | 1.10 | 1.00 | 0.91 | 0.82 | |
| use of software tools | 1.24 | 1.10 | 1.00 | 0.91 | 0.83 | |
| required development schedule | 1.23 | 1.08 | 1.00 | 1.04 | 1.10 | |

# Example Intermediate CoCoMo

Consider a project to develop a full screen editor. The major components identified are (1) Screen edit, (2) Command Language Interpreter, (3) File input and output, (4) Cursor movement and (5) Screen movement. The sizes for these are estimated to be 4K, 2K, 1K, 2K and 3K delivered source code lines. Use COCOMO model to determine:

(a) Overall cost and schedule estimates (assume values for different cost drivers, with at least three of them being different from 1.0).

(b) Cost and schedule estimates for different phases.

4 2 2 3 1

12K

On Tuesday 6th April , your job is to

(a) Identify Modules from given project

(b) Get estimates of each module

?

UNIVERSITY OF KARACHI

# Solution – Step I

Size of 5 modules are:-

| | |
|---|---|
| Screen edit | = 4KLOC |
| Command Language Interpreter | = 2KLOC |
| File input and output | = 1KLOC |
| Cursor movement and | = 2KLOC |
| Screen movement | = 3KLOC |
| **total** | + **= 12KLOC** |

# Solution – Step II

PA

| | *Description* | *Very Low* | *Low* | *Nominal* | *High* | *Very High* | *Extra High* |
|---|---|---|---|---|---|---|---|
| **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 |

CA

| | *Description* | *Very Low* | *Low* | *Nominal* | *High* | *Very High* | *Extra High* |
|---|---|---|---|---|---|---|---|
| **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 | - |

# Solution Step III

PA

| | Description | Very Low | Low | Nominal | High | Very High | Extra High |
|---|---|---|---|---|---|---|---|
| **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 | - | - |

PA

| | Description | Very Low | Low | Nominal | High | Very High | Extra High |
|---|---|---|---|---|---|---|---|
| **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 | - |

UNIVERSITY OF KARACHI

# Solution Step IV calculate EAF

Let us assume that significant cost drivers are

| | |
|---|---|
| (1) Required software reliability is high i.e. | 1.15 |
| (2) Product complexity is high i.e. | 1.15 |
| (3) Analyst capability is high i.e. | 0.86 |
| (4) All other drivers are nominal i.e. | 1.00 |

Hence

$$EAF = 1.15 * 1.15 * 0.86 = 1.1373$$

UNIVERSITY OF KARACHI

# Solution Step V Estimate E and D

$$E = a_i\,(KLOC)^{b_i} * EAF$$
$$= 3.2(12)^{1.05} * 1.1373 = 49.449 \text{ PM}$$

$$D = c_i\,(E)^{d_i}$$
$$= 2.5(49.44)^{0.38} = 11.007 \text{ M}$$

UNIVERSITY OF KARACHI

# Solution Step VI Phase Estimates

| Mode and code size | Plan and requirement | System design | Detail design | Module code and test | Integration and test |
|---|---|---|---|---|---|
| **Lifecycle Phase Value of $\mu_b$** E | | | | | |
| Organic Small S ≈ 2 | 0.06 | 0.16 | 0.26 | 0.42 | 0.16 |
| Organic Medium S ≈ 32 | 0.06 | 0.16 | 0.24 | 0.38 | 0.22 |
| Semidetached Medium S ≈ 32 | 0.07 | 0.17 | 0.25 | 0.33 | 0.25 |
| Semidetached Large S ≈ 128 | 0.07 | 0.17 | 0.24 | 0.31 | 0.28 |
| Embedded Large S ≈ 128 | 0.08 | 0.18 | 0.25 | 0.26 | 0.31 |
| Embedded Extra Large S ≈ 320 | 0.08 | 0.18 | 0.24 | 0.24 | 0.34 |
| **Lifecycle Phase Value of $\sqcup_b$** T / D | | | | | |
| Organic Small S ≈ 2 | 0.10 | 0.19 | 0.24 | 0.39 | 0.18 |
| Organic Medium S ≈ 32 | 0.12 | 0.19 | 0.21 | 0.34 | 0.26 |
| Semidetached Medium S ≈ 32 | 0.20 | 0.26 | 0.21 | 0.27 | 0.26 |
| Semidetached Large S ≈ 128 | 0.22 | 0.27 | 0.19 | 0.25 | 0.29 |
| Embedded Large S ≈ 128 | 0.36 | 0.36 | 0.18 | 0.18 | 0.28 |
| Embedded Extra Large S ≈ 320 | 0.40 | 0.38 | 0.16 | 0.16 | 0.30 |

# Phase wise Effort Estimates

Phase wise cost and schedule estimates

$$E_p = \mu_p E$$

$$D_p = \square_p D$$

Since size is only 12 KLOC, it is an organic small model.

Phase wise effort distribution is given below:

| | |
|---|---|
| System Design | $= 0.16 * 49.449 = 7.911$ |
| Detailed Design | $= 0.26 * 49.449 = 12.856$ |
| Module code and test | $= 0.42 * 49.449 = 20.768$ |
| Integration and test | $= 0.16 * 49.449 = 7.911$ |

UNIVERSITY OF KARACHI

# Phase wise Time Estimates

Phase wise development time duration is:

| | |
|---|---|
| System Design | = 0.19 * 11.007 = 2.091 |
| Detailed Design | = 0.24 * 11.007 = 2.641 |
| Module code and test | = 0.39 * 11.007 = 4.292 |
| Integration and test | = 0.18 * 11.007 = 1.981 |

# COCOMO II - Constructive Cost Model (softwarecost.org)

Not secure | softwarecost.org/tools/COCOMO/

**COCOMO II - Constructive Cost Model**

**Software Size** Sizing Method [Source Lines of Code ▾]

| | SLOC | % Design Modified | % Code Modified | % Integration Required | Assessment and Assimilation (0% - 8%) | Software Understanding (0% - 50%) | Unfamiliarity (0-1) |
|---|---|---|---|---|---|---|---|
| New | | | | | | | |
| Reused | | 0 | 0 | | | | |
| Modified | | | | | | | |

**Software Scale Drivers**

| | | | | | |
|---|---|---|---|---|---|
| Precedentedness | Nominal | Architecture / Risk Resolution | Nominal | Process Maturity | Nominal |
| Development Flexibility | Nominal | Team Cohesion | Nominal | | |

**Software Cost Drivers**

**Product**

| | |
|---|---|
| Required Software Reliability | Nominal |
| Data Base Size | Nominal |
| Product Complexity | Nominal |
| Developed for Reusability | Nominal |
| Documentation Match to Lifecycle Needs | Nominal |

**Personnel**

| | |
|---|---|
| Analyst Capability | Nominal |
| Programmer Capability | Nominal |
| Personnel Continuity | Nominal |
| Application Experience | Nominal |
| Platform Experience | Nominal |
| Language and Toolset Experience | Nominal |

**Platform**

| | |
|---|---|
| Time Constraint | Nominal |
| Storage Constraint | Nominal |
| Platform Volatility | Nominal |

**Project**

| | |
|---|---|
| Use of Software Tools | Nominal |
| Multisite Development | Nominal |
| Required Development Schedule | Nominal |

**Maintenance** [Off ▾]

**Software Labor Rates**

Cost per Person-Month (Dollars) [ ]

[Calculate]

# COCOMO II Challenges

1995: one-size-fits-all model for 21st century software

1999: poor fit for schedule-optimized projects;
CORADMO

2000: poor fit for COTS-intensive projects: COCOTS

2003: need model for product line investment: COPLIMO

2003: poor fit for agile projects: Agile COCOMO II
(partial)

2012: poor fit for incremental development: COINCOMO

UNIVERSITY OF KARACHI

# Functional Point

[Basic COCOMO Model (umich.edu)]

# Functional Point Formula

Functions points are obtained by multiplying Unadjusted Functional Points by Value Adjustment Factors

$$\boxed{FP = UFP * VAF}$$

$UFP$ = Unadjusted Functional Points

VAF = Value Adjustment Factor

# VAF = Value Adjustment Factor

| General System Characteristics | WEIGHT |
|---|---|
| 01. Data Communications | 5 |
| 02. Distributed Data Processing | 4 |
| 03. Performance | 3 |
| 04. Heavily Used Configuration | 2 |
| 05. Transaction Rate | 3 |
| 06. On-line Data Entry | 5 |
| 07. End-User Efficiency | 4 |
| 08. On-line Update | 5 |
| 09. Complex Processing | 2 |
| 10. Reusability | 3 |
| 11. Installation Ease | 1 |
| 12. Operational Ease | 3 |
| 13. Multiple Sites | 2 |
| 14. Facilitate Change | 5 |
| Total Degrees of Influence (TDI): | 47 |
| Value Adjustment Factor (VAF): | 1.12 |

Evaluate each of the 14 GSCs on a scale from 0 -5 to determine the degree of influence (DI).

Calculate the degrees of influence to produce total degree of influence (TDI).

Insert the TDI into the formula to produce the VAF

| Score As | System Influence |
|---|---|
| 0 | Not Present or no influence |
| 1 | Incidental influence |
| 2 | Moderate influence |
| 3 | Average influence |
| 4 | Significant influence |
| 5 | Strong influence throughout |

Formula: VAF=(TDI*0.01)+0.65

UNIVERSITY OF KARACHI

# Complexity Table Gives UFP

| Information Domain Value | Count | | Weighting factor Simple | Average | Complex | | |
|---|---|---|---|---|---|---|---|
| External Inputs (EIs) | | × | 3 | 4 | 6 | = | |
| External Outputs (EOs) | | × | 4 | 5 | 7 | = | |
| External Inquiries (EQs) | | × | 3 | 4 | 6 | = | |
| Internal Logical Files (ILFs) | | × | 7 | 10 | 15 | = | |
| External Interface Files (EIFs) | | × | 5 | 7 | 10 | = | |
| Count total | | | | | | → | |

# Final Computation of FP

| Information Domain Value | Count | | Weighting factor: Simple | Average | Complex | | |
|---|---|---|---|---|---|---|---|
| External Inputs (EIs) | 3 | × | 3 | 4 | 6 | = | 9 |
| External Outputs (EOs) | 2 | × | 4 | 5 | 7 | = | 8 |
| External Inquiries (EQs) | 2 | × | 3 | 4 | 6 | = | 6 |
| Internal Logical Files (ILFs) | 1 | × | 7 | 10 | 15 | = | 7 |
| External Interface Files (EIFs) | 4 | × | 5 | 7 | 10 | = | 20 |
| Count total | | | | | | → | 50 |

The count total shown in Figure above must be adjusted using Equation above. For the purposes of this example, we assume that Σ(Fi ) is 46 (a moderately complex product). Therefore,

FP =50 *[0.65 + (0.01* 46)]= 56.

2

# FP increases with requirement change

Requirements → Functional Design → Detail Design

100 FPs → 120 FPs → 130 FPs → 135 FPs

UNIVERSITY OF KARACHI

# Impact of changed FP

| Impact | | | |
|---|---|---|---|
| Effort | + 1 month | + .5 month | + .25 month |
| Schedule | + 2 weeks | + 1 week | + 2.5 days |
| Cost | + $5 K | + $2.5 K | + $1.25 K |

Source: International Function Point User Group 2001

# Past FP Analysis

| | | |
|---|---|---|
| 1 | Oracle | 229,434 |
| 2 | Windows 7 (all features) | 202,150 |
| 3 | Windows XP | 66,238 |
| 4 | Google Docs | 47,668 |
| 5 | Microsoft Office 2003 | 33,736 |
| 6 | F15 Avionics / Weapons | 23,109 |
| 7 | Apple iPhone | 19,366 |
| 8 | Google Search Engine | 18,640 |
| 9 | Linux | 17,505 |
| 10 | Facebook | 8,404 |
| 11 | MapQuest | 3,793 |
| 12 | Microsoft Project | 1,963 |
| 13 | Google Android OS (Original Version | 1,858 |
| 14 | Mozilla Firefox | 1,342 |
| 15 | Java Compiler | 1,185 |
| 16 | Wikipedia | 1,142 |
| 17 | Twitter (Original 2009) | 541 |

# Past FP Analysis

| Applications | Approximate Size in Function Points |
|---|---|
| Star Wars Missile Defense | 350,000 |
| ERP (SAP, Oracle etc.,) | 300,000 |
| Microsoft Windows Vista | 159,000 |
| Microsoft Office 2007 | 98,000 |
| Airline Reservation System | 50,000 |
| NASA Space shuttle | 25,000 |

UNIVERSITY OF KARACHI

Department of Compute Science (UBIT Building), Karachi, Pakistan.

1200 Acres (5.2 Km sq.)
53 Departments
19 Institutes
25000 Students

UNIVERSITY OF KARACHI