# Software Project Estimation

# Cost Estimation

- project scope must be explicitly defined
- task and/or functional decomposition is necessary
- historical measures (metrics) are very helpful
- at least two different techniques should be used
- remember that uncertainty is inherent

# To Understand Scope …

- Understand the customers needs
- understand the business context
- understand the project boundaries
- understand the customer's motivation
- understand the likely paths for change
- understand that ...

*Even when you understand, nothing is guaranteed!*

# Estimation Techniques

- past (similar) project experience
- conventional estimation techniques
  - task breakdown and effort estimates
  - size (e.g., FP) estimates
- tools

# Functional Decomposition

**Statement of Scope**

perform a "grammatical parse"

**functional decomposition**
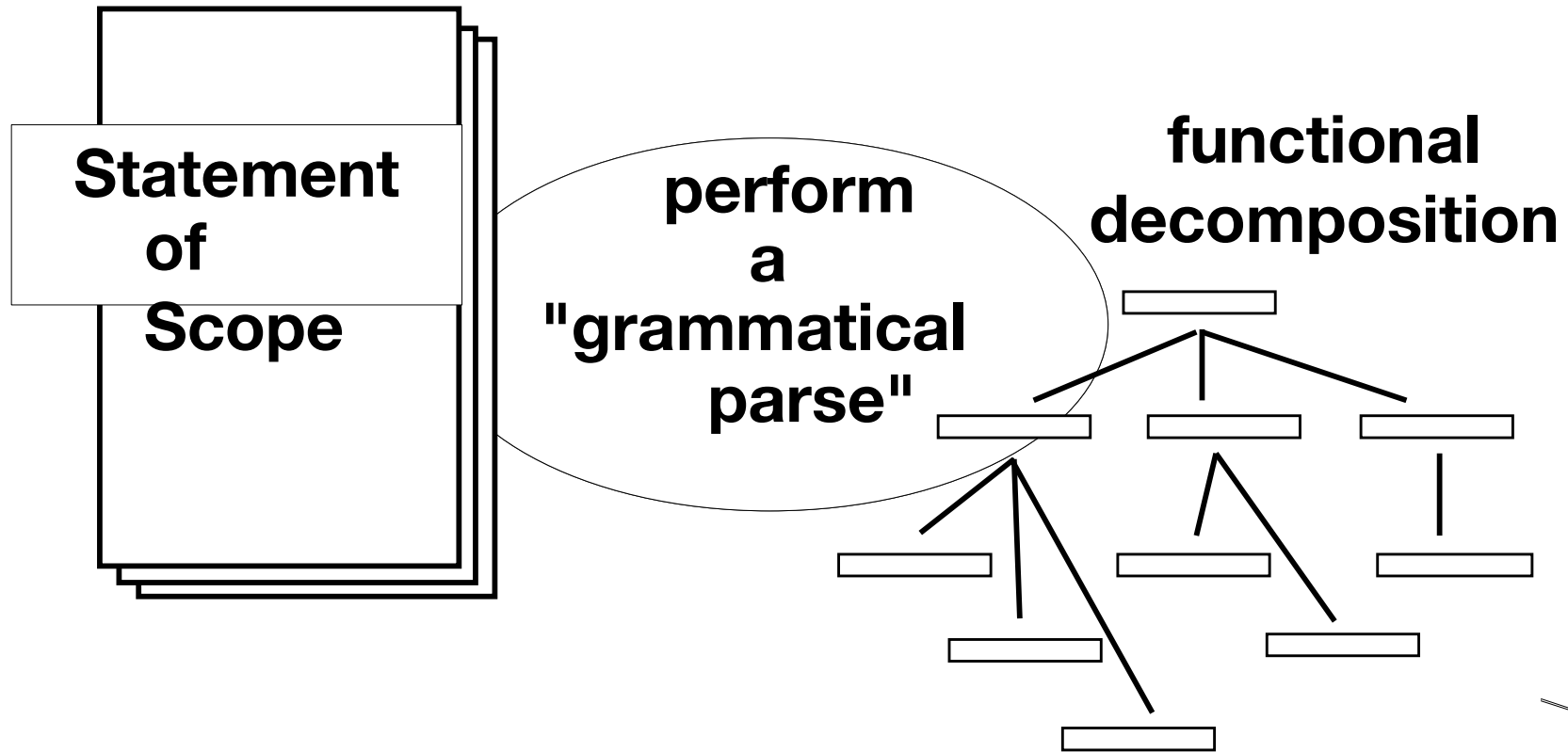
# Conventional Methods: LOC/FP Approach

- compute LOC/FP using estimates of information domain values
- use historical effort for the project

# Example: LOC Approach

| Functions | estimated LOC | LOC/pm | $/LOC | Cost | Effort (months) |
|-----------|---------------|--------|-------|------|-----------------|
| UICF | 2340 | 315 | 14 | 32,000 | 7.4 |
| 2DGA | 5380 | 220 | 20 | 107,000 | 24.4 |
| 3DGA | 6800 | 220 | 20 | 136,000 | 30.9 |
| DSM | 3350 | 240 | 18 | 60,000 | 13.9 |
| CGDF | 4950 | 200 | 22 | 109,000 | 24.7 |
| PCF | 2140 | 140 | 28 | 60,000 | 15.2 |
| DAM | 8400 | 300 | 18 | 151,000 | 28.0 |
| Totals | 33,360 | | | 655,000 | 145.0 |

# Example: FP Approach

| measurement parameter | count | | weight | | |
|---|---|---|---|---|---|
| number of user inputs | 40 | x | 4 | = | 160 |
| number of user outputs | 25 | x | 5 | = | 125 |
| number of user inquiries | 12 | x | 4 | = | 48 |
| number of files | 4 | x | 7 | = | 28 |
| number of ext.interfaces | 4 | x | 7 | = | 28 |
| algorithms | 60 | x | 3 | = | 180 |
| count-total | | | | | 569 |
| complexity multiplier | | | | | .84 |
| feature points | | | | | 478 |

$\times$ $\boxed{0.25 \text{ p-m / FP}}$ = 120 p-m

# Creating a Task Matrix

**Obtained from "process framework"**

| | framework activities |
|---|---|
| application functions | Effort required to accomplish each framework activity for each application function |
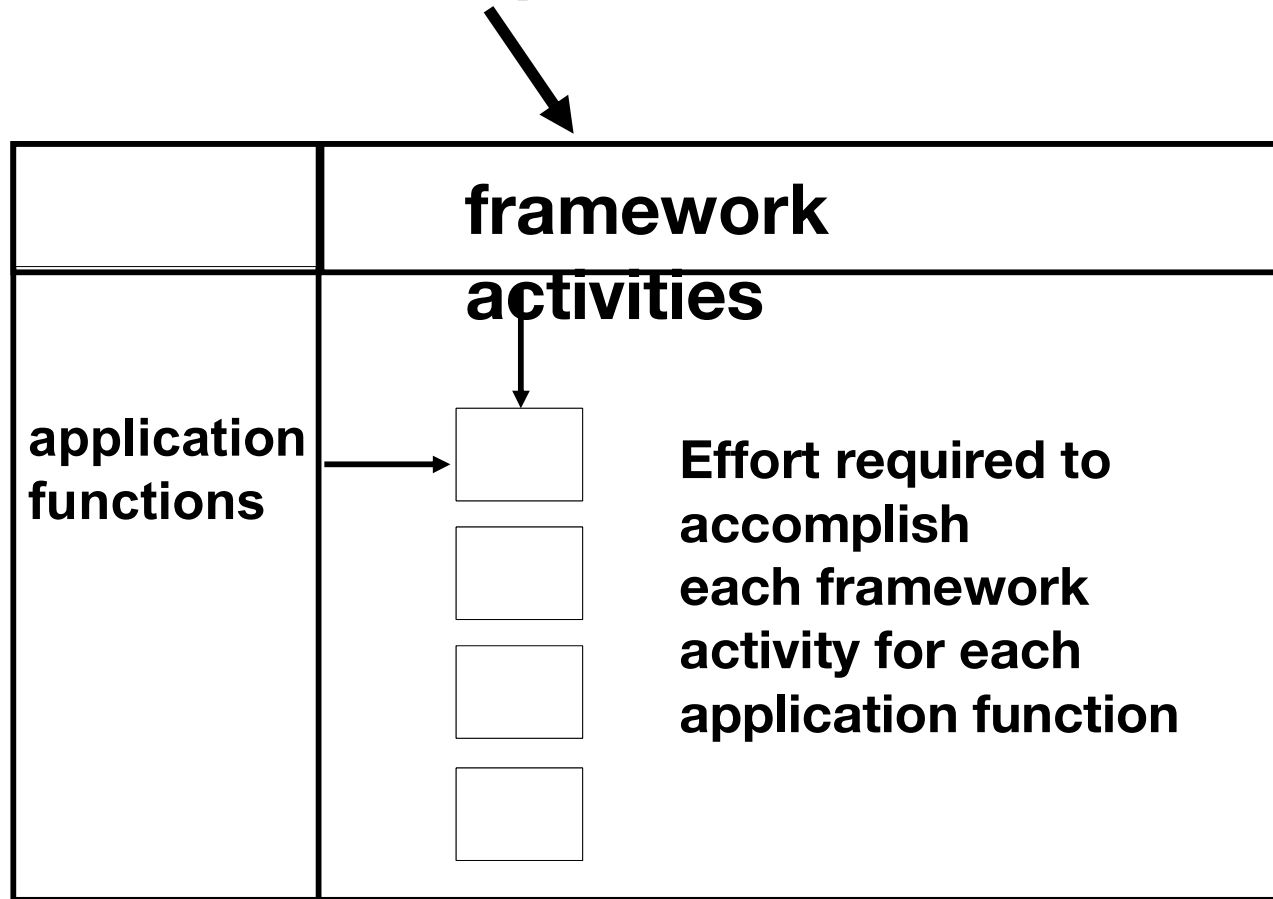
# Empirical Estimation Models

*General form:*

effort = tuning coefficient * size          exponent

**usually derived
as person-
months
of effort
required**

**a number derived based
either a constant
or
on complexity of
project**

**usually LOC but
may also be
function point**

**empirically
derived**
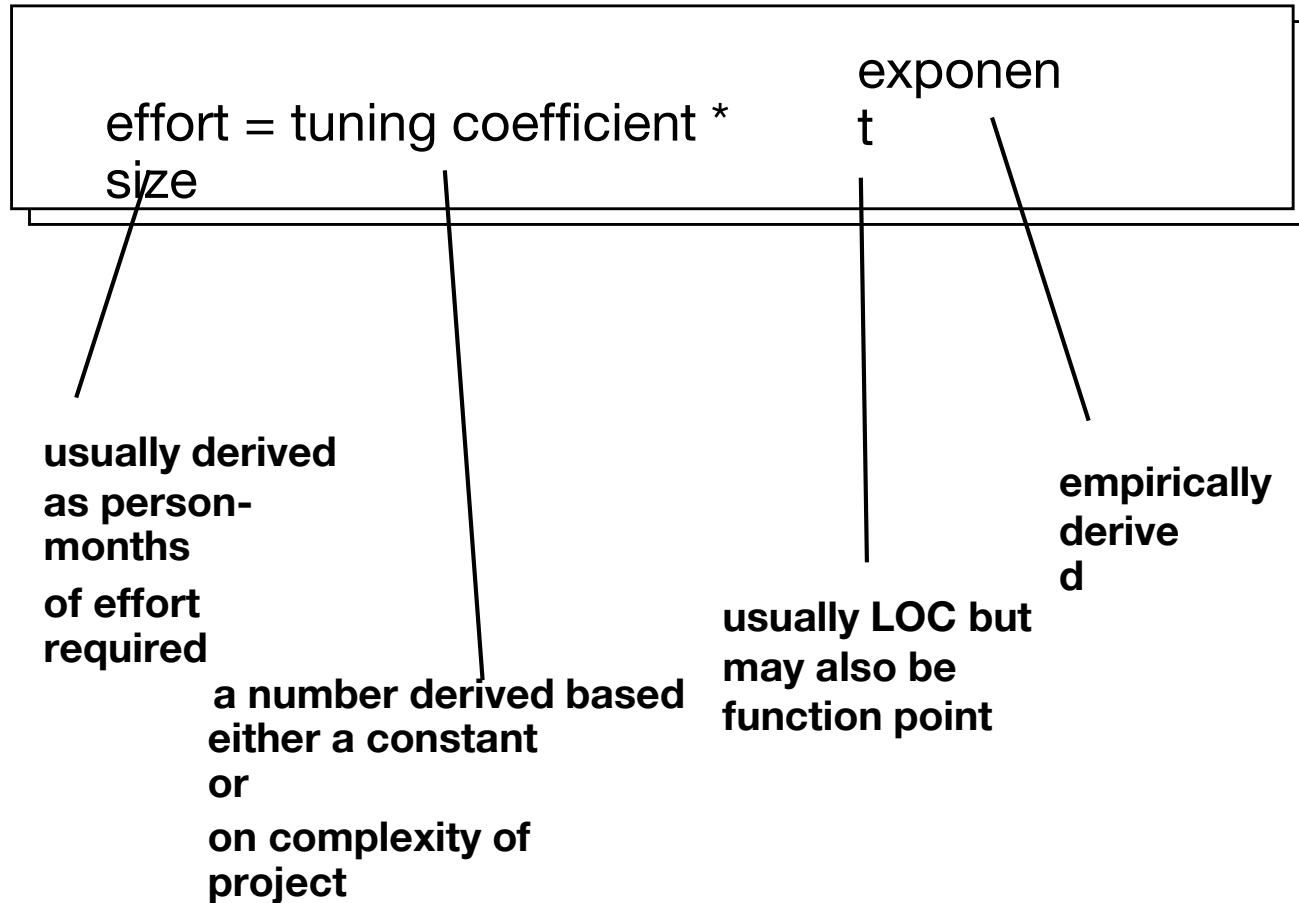
# Empirical Estimation Model

## LOC-Oriented Estimation models

$E = 5.2 \times (KLOC)^{.91}$          Waltson-Felix Model

$E = 5.5 + 0.73 \times (KLOC)^{1.16}$        Bailey-Basili model

$E = 3.2 \times (KLOC)^{1.05}$          Boehm Simplw Model

## FP-Oriented Estimation models

$E = -13.39 \times .0545\ FP$        Albrecht and Gaffney Model

$E = 60.62 \times 7.728 \times 10\text{-}8\ FP^3$      Kemerer model

$E = 585.7 \times 15.12\ FP$         Matson, Barnett, and Mellichamo Model

# COCOMO Model

- Bery Boehm introduced a hierarchy of s/w estimation models bearing the name COCOMO, for COnstructive COst MOdel.
  - Empirical models for estimating effort and time

See : sunset.usc.edu/COCOMOII/cocomo.html

# COCOMO II

COCOMO II is actually a hierarchy of estimation models that address the following areas:

- Application Composition model
- Early Design Stage Model
- Reuse Model
- Post-architecture stage model

# Object Points Estimate

COCOMO II application composition model uses object points

Like Function Points, the Object Point is an indirect s/w measure that is computed using counts of number of:

    1) Screens (UI)
    2) Reports
    3) Components

**NOP = Object Points x [ (100-%reuse)/100]**

**PORD is a parameter with unit NOP / person-month**

**Estimated effort = NOP / PROD**

# Early Design Model

Requirement Analysis done

You can estimate the size of code by LOC

$PM = A \times Size^B \times M$

Where

$M = PERS \times RCPX \times RUSE \times PDIF \times PREX \times FCIL \times SCED$

**A =2.94, B = 1.1 to 1.24 depending n the novelty**

**PERS= Personnel Capability**

**RCPX= Product reliability and Complexity**

**RUSE= Reuse required**

**PDIF = Platform difficulty**

**PREX = Personnel Experience**

**SCED= Required Schedule**

**FCIL = Team support facility**

# Reuse Mode

Black Box Reuse: The component will be reused with necessary configuration

White Box Reuse: Code will be modified when reused The number of new code will be estimated

For generated code

$PM = (ASLOC * AT/100)/ ATPROD$

ASLOC: Number of lines of Generated Code

AT: Percentage of Code Automatically Generated

ATPROD: Productivity of Engineers to generate code

When code is understood for integration

$ESLOC = ASLOC * (1-AT/100) * AAM$

AAM= adaptation adjustment multiplier

From the cost of changing the reused code

# Post Architecture Stage Model

Same as early design model
17 rather than 7 multipliers
Code size will be estimated
  Number of new lines of codes
  Modified as there might be change of requirements
Exponent term depends on the following factors (ranked
  between 1 to 5, 1 for excellent, 5 for poor),
  Precedenteness
  Development flexibility
  Architecture/Risk Resolution
  Team Cohesion
  Process maturity
Sum/100 +1.01 is the exponent term

# The Software Equation

- It is a dynamic multivariable model that assumes a specific distribution of effort over a life of s/w development of project. [from 4000 s/ projects]

$$E = [LOC \times B^{0.333} /P ]^3 \times (1/t^4)$$

E = effort in person-months/person-years

t = project duration in months or years

B =special skill factors [ B =.16 (5-15 KLOC)

B = .39 (70 kloc)]

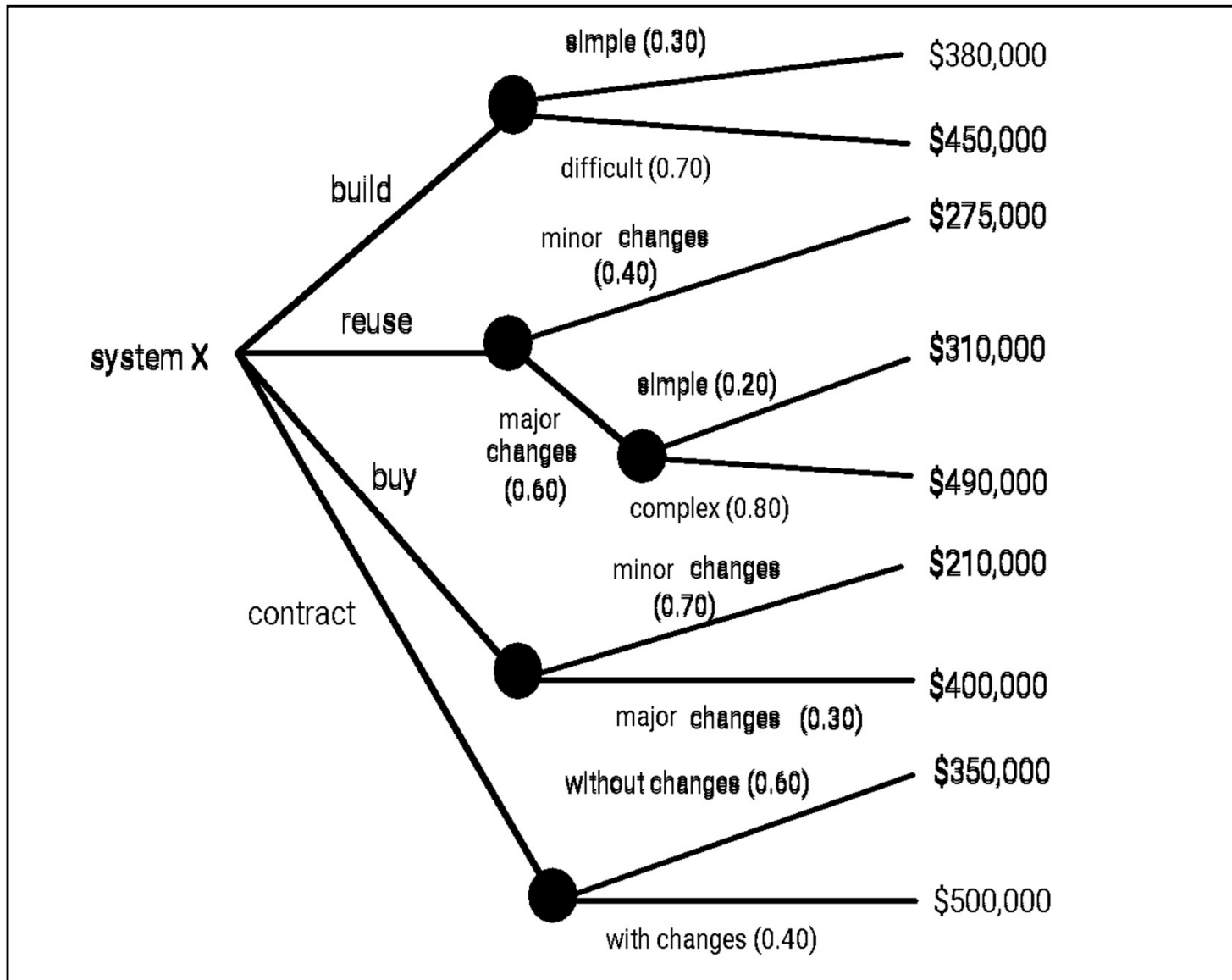P = productivity parameter [ 2000 -> embedded s/w

10,000 -> telecomm

20,000 -> business app]

# Estimation Guidelines

- estimate using at least two techniques
- get estimates from independent sources
- avoid over-optimism, assume difficulties

- you've arrived at an estimate, sleep on it
- adjust for the people who'll be doing the job—they have the highest impact

# The Make-Buy Decision

# Computing Expected Cost

expected cost

$$= \sum_i (\text{path probability})_i \times (\text{estimated path cost})_i$$

*For example, the expected cost to build is:*

expected cost$_{build}$ = 0.30($380K)+0.70($450K)
= $429 K

*similarly,*

expected cost$_{reuse}$ = $382K

expected cost$_{buy}$ = $267K

expected cost$_{contr}$ = $410K