Module 3 : Software Project Planning & Cost Estimation

Mrs. Priya R L
Faculty Incharge for PM
Department of Computer Engineering
VES Institute of Technology, Mumbai

ILOC: Project Management (Autonomy)

V.E.S.
Since 1962

University of Mumbai

Agenda : Software Cost Estimation

- Software Size Estimation
- Function Point (Numericals)
- Cost Estimation: COCOMO (Numericals),
- COCOMO-II (Numericals) till Early design model.

Software Size Estimation

- **Predicting the resources required** for a software development process.
- How much effort is required to complete an activity?
- How much calendar time is needed to complete an activity?
- What is the total cost of an activity?
- Project estimation and scheduling and interleaved management activities.

Software Cost Components

- Hardware and software costs
- Travel and training costs
- Effort costs (the dominant factor in most projects)
  - salaries of engineers involved in the project
  - Social and insurance costs
- Effort costs must take overheads into account
  - costs of building, heating, lighting
  - costs of networking and communications
  - costs of shared facilities (e.g library, staff restaurant, etc.)

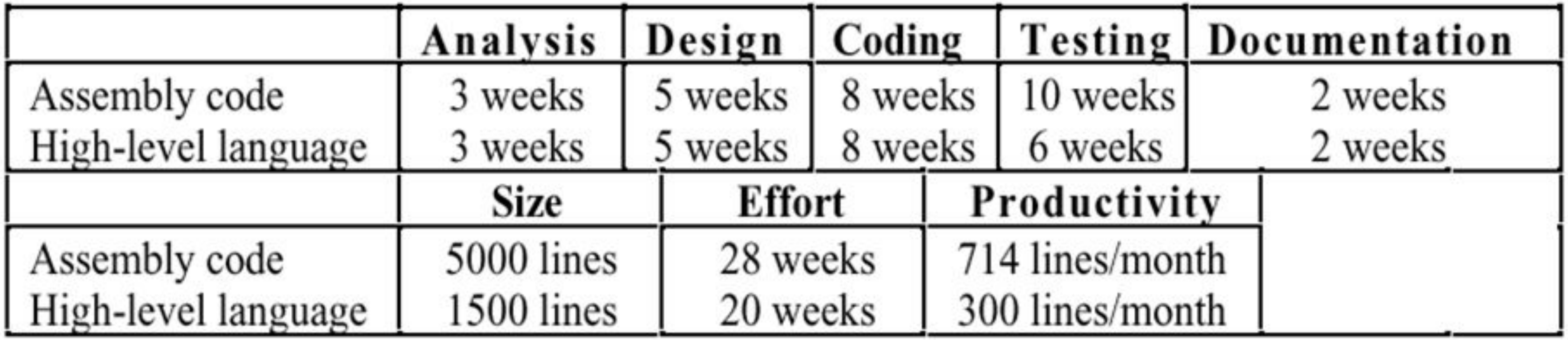
Software Estimation - Productivity Measures

- **Size related measures** based on some output from the software process. This may be lines of delivered source code, object code instructions, etc.
- **Function-related measures** based on an estimate of the functionality of the delivered software. Function-points are the best known of this type of measure

Software Measurement Problems

- Estimating the size of the measure
- Estimating the total number of programmer months which have elapsed
- Estimating contractor productivity (e.g. documentation team) and incorporating this
- Estimate in overall estimate

Lines of Code (LOC) Method



	Analysis	Design	Coding	Testing	Documentation
Assembly code	3 weeks	5 weeks	8 weeks	10 weeks	2 weeks
High-level language	3 weeks	5 weeks	8 weeks	6 weeks	2 weeks

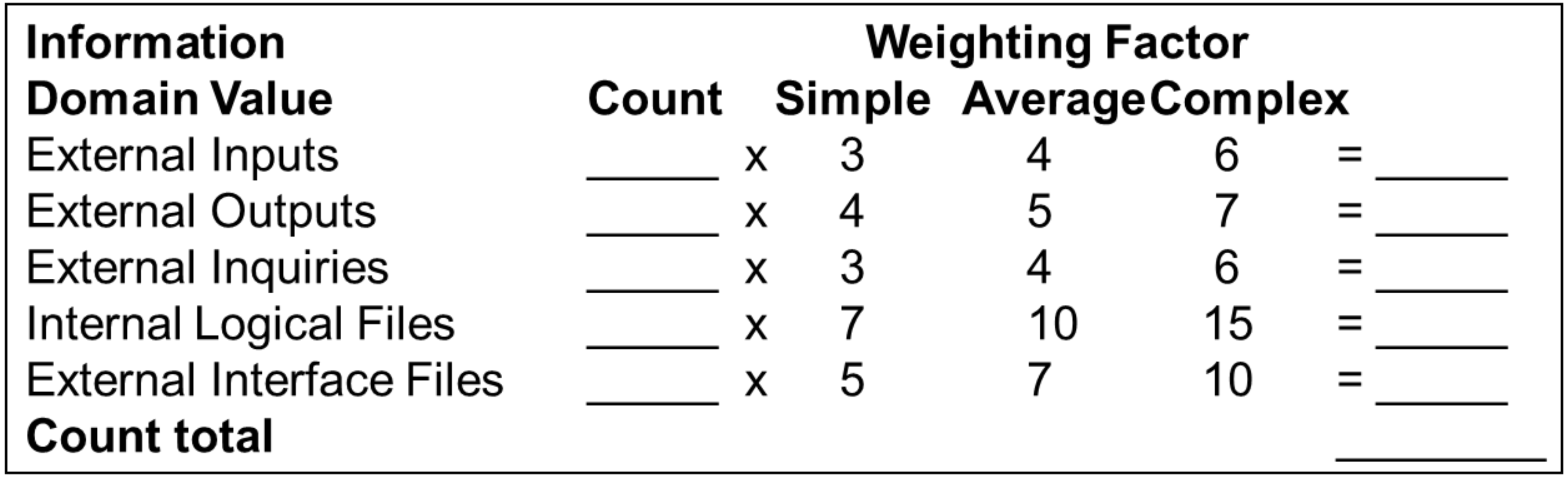
	Size	Effort	Productivity
Assembly code	5000 lines	28 weeks	714 lines/month
High-level language	1500 lines	20 weeks	300 lines/month

Function Point Analysis (FPA)

- It is designed to estimate and measure the cost and effort of developing new software applications and maintaining existing software applications.
- It was developed by A.J. Albrecht of the IBM Corporation in the early 1980s.
- The main other approach used for measuring the size of software project is lines of code (LOC) – which has a number of inherent problems.

How FPA done?

- Identify / collect the **information domain values**
- Complete the table shown below to **get the count total**
- **Associate a weighting factor** (i.e., complexity value) with each count based on criteria established by the software development organization



Information			Weighting Factor				
Domain Value	**Count**		**Simple**	**Average**	**Complex**		
External Inputs	______	x	3	4	6	=	______
External Outputs	______	x	4	5	7	=	______
External Inquiries	______	x	3	4	6	=	______
Internal Logical Files	______	x	7	10	15	=	______
External Interface Files	______	x	5	7	10	=	______
Count total							______

ILOC: Project Management (Autonomy)

How FPA done?

A simple example:

inputs

3 simple X 3 = 9

4 average X 4 = 16

1 complex X 6 = 6

outputs

6 average X 5 = 30

2 complex X 7 = 14

internal logical files

5 complex X 15 = 75

inquiries

8 average X 4 = 32

interfaces

3 average X 7 = 21

4 complex X 10 = <u>40</u>

Unadjusted function points 243

How FPA done?

- Evaluate and sum up the adjustment factors (see the next two slides)
- **“F_i” refers to 14 value adjustment factors,** with each ranging in value from 0 (not important) to 5 (absolutely essential)

Value Adjustment Factors

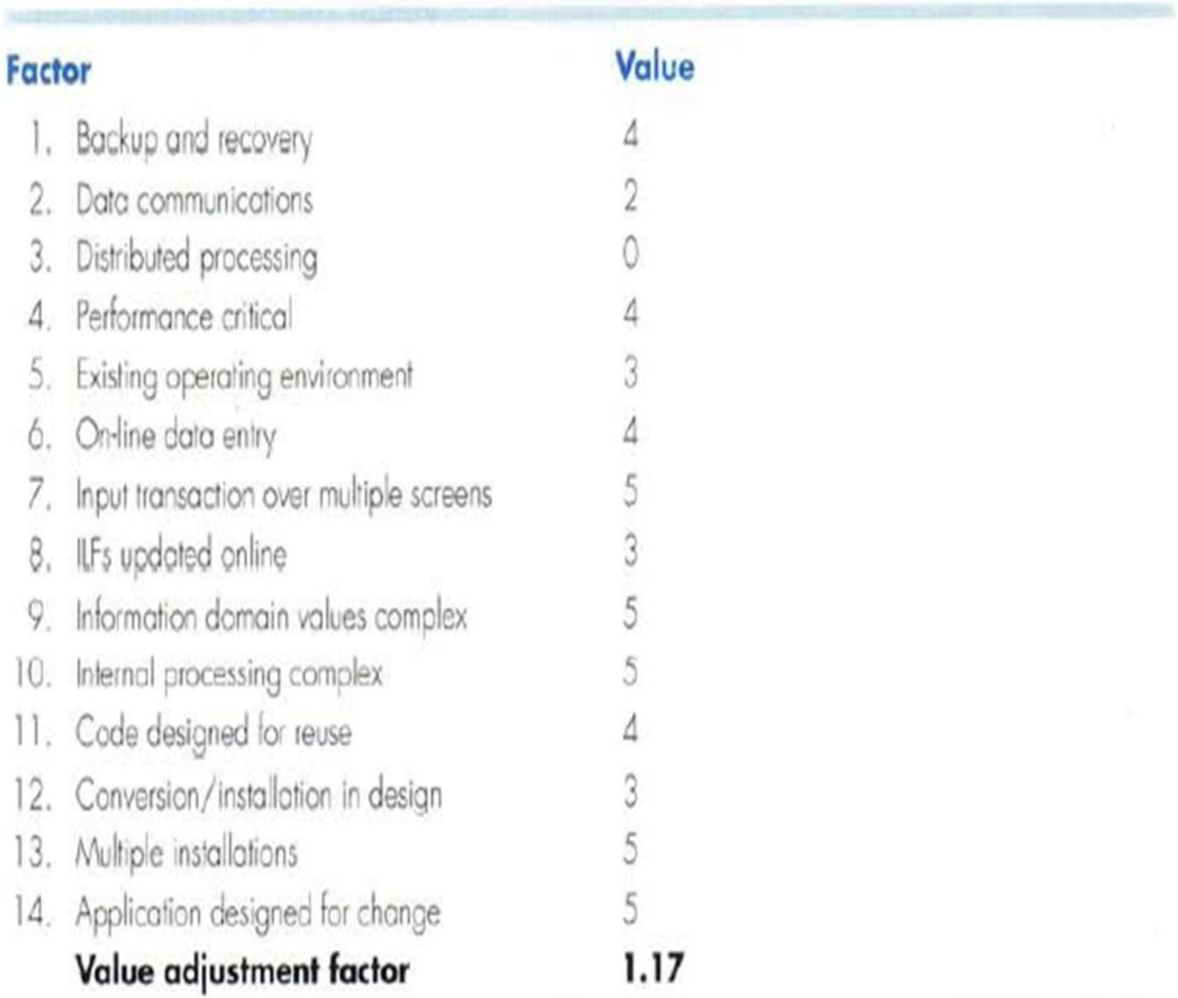
1) Does the system require reliable backup and recovery?
2) Are specialized data communications required to transfer information to or from the application?
3) Are there distributed processing functions?
4) Is performance critical?
5) Will the system run in an existing, heavily utilized operational environment?
6) Does the system require online data entry?
7) Does the online data entry require the input transaction to be built over multiple screens or operations?

Value Adjustment Factors

8) Are the internal logical files updated on-line?
9) Are the inputs, outputs, files, or inquiries complex?
10) Is the internal processing complex?
11) Is the code designed to be reusable?
12) Are conversion and installation included in the design?
13) Is the system designed for multiple installations in different organizations?
14) Is the application designed to facilitate change and for ease of use by the user?

V.E.S.
Since 1962

Value Adjustment Factors



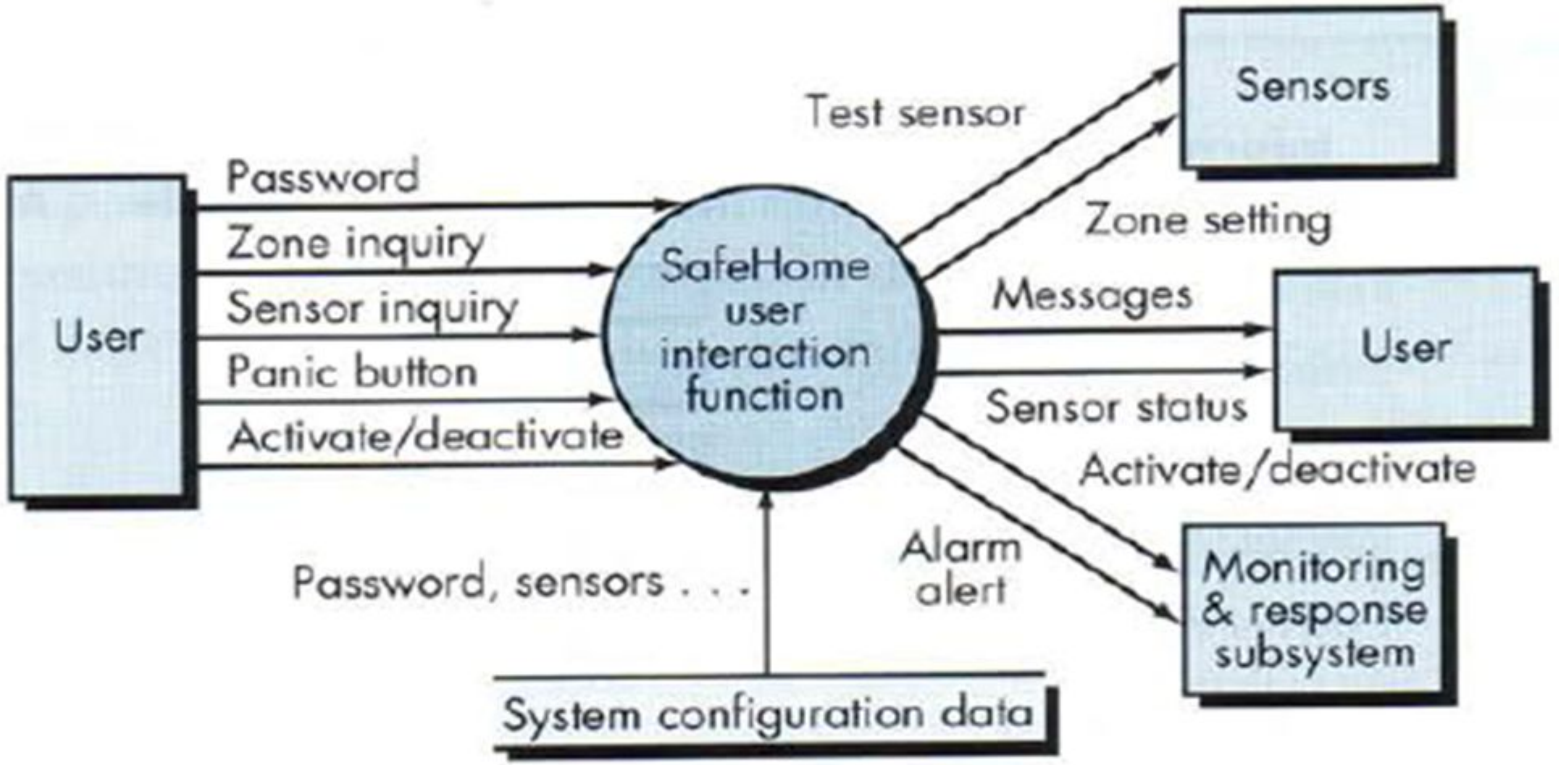
Factor	Value
1. Backup and recovery	4
2. Data communications	2
3. Distributed processing	0
4. Performance critical	4
5. Existing operating environment	3
6. On-line data entry	4
7. Input transaction over multiple screens	5
8. ILFs updated online	3
9. Information domain values complex	5
10. Internal processing complex	5
11. Code designed for reuse	4
12. Conversion/installation in design	3
13. Multiple installations	5
14. Application designed for change	5
Value adjustment factor	**1.17**

FPA Calculations

Compute the number of function points (FP)

$$FP = \text{count total} * [0.65 + 0.01*\text{sum}(F_i)]$$

FPA : Example

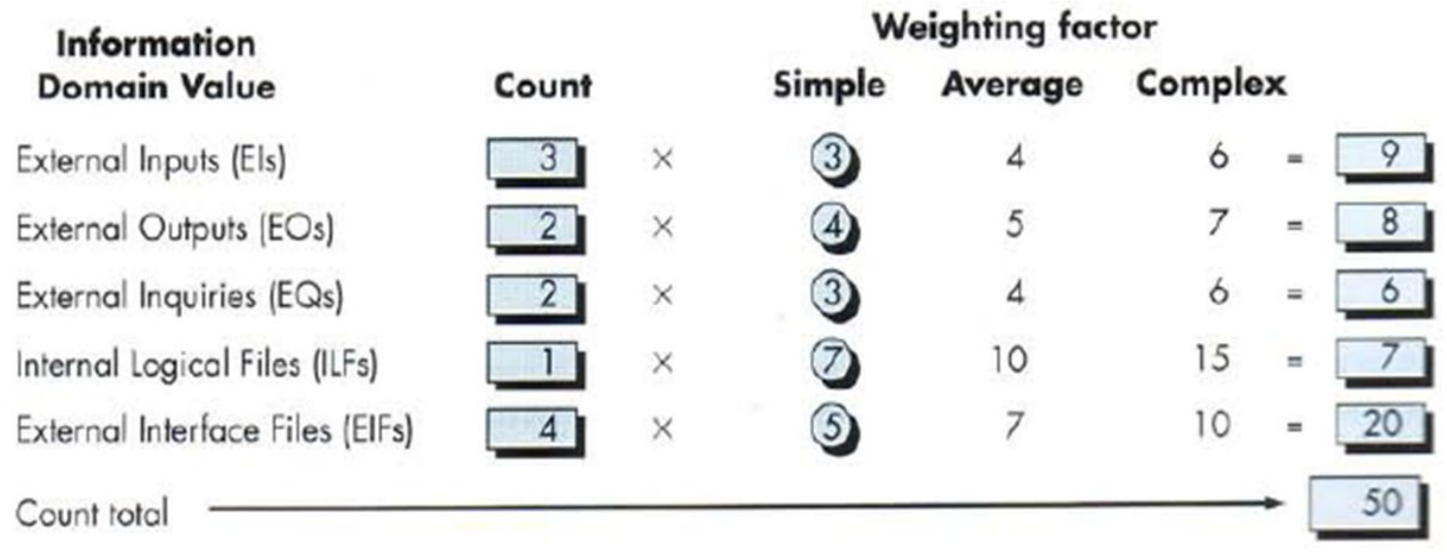


Test sensor
Sensors
Password
Zone inquiry
Sensor inquiry
Panic button
Activate/deactivate
User
SafeHome user interaction function
Zone setting
Messages
User
Sensor status
Activate/deactivate
Alarm alert
Monitoring & response subsystem
Password, sensors . . .
System configuration data

FPA : Example

- **External Inputs (EIs) :** Password, Panic Button, Activate/Deactivate
- **External Outputs(EOs):** Messages and Sensors
- **External Inquiries (EQs):** Zone and Sensor Inquiry
- **Internal Logical Files(ILFs):** System Configuration File
- **External Interface Files(EIFs):** Test Sensor, Zone setting, Activate, deactivate and Alarm alert

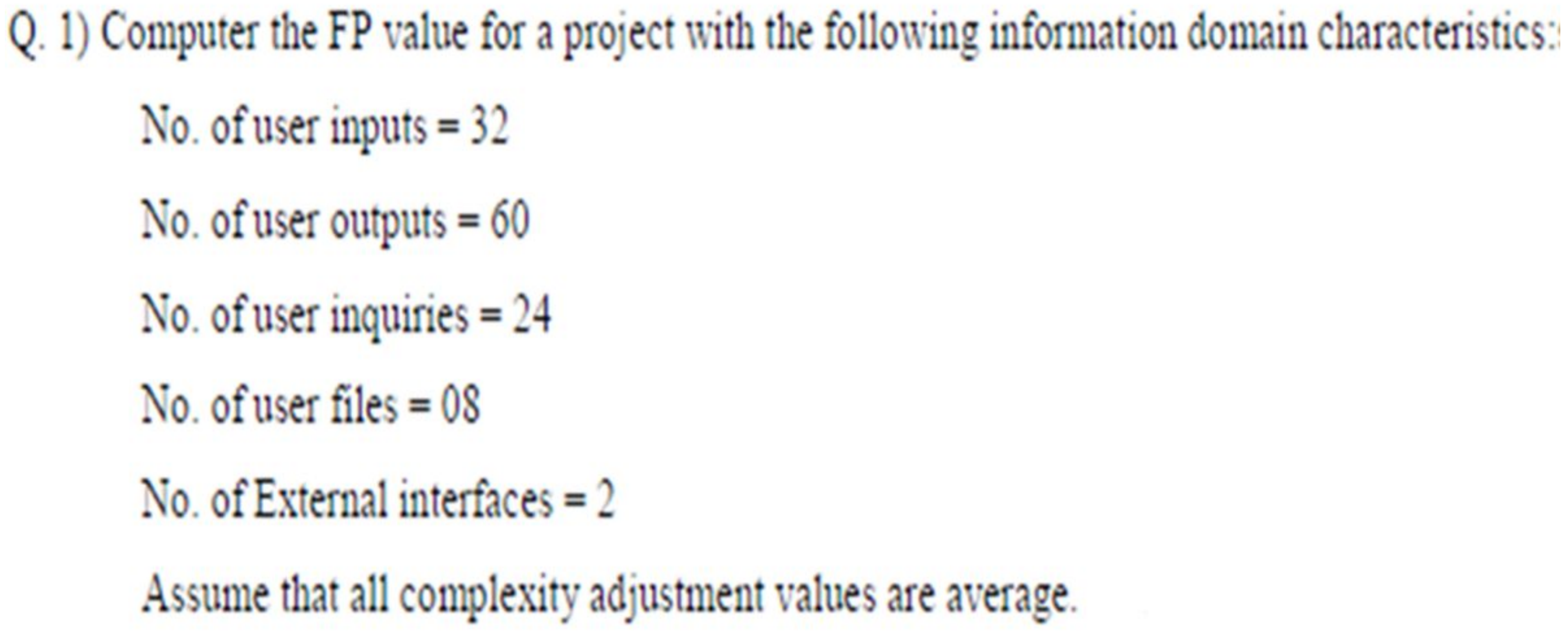
FPA : Example



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

- FP = count total * [0.65 + 0.01 * sum(F_i)]
- FP = 50 * [0.65 + (0.01 * 46)]
- FP = 55.5 (rounded up to 56)

FPA : Exercise



Q. 1) Computer the FP value for a project with the following information domain characteristics:

No. of user inputs = 32

No. of user outputs = 60

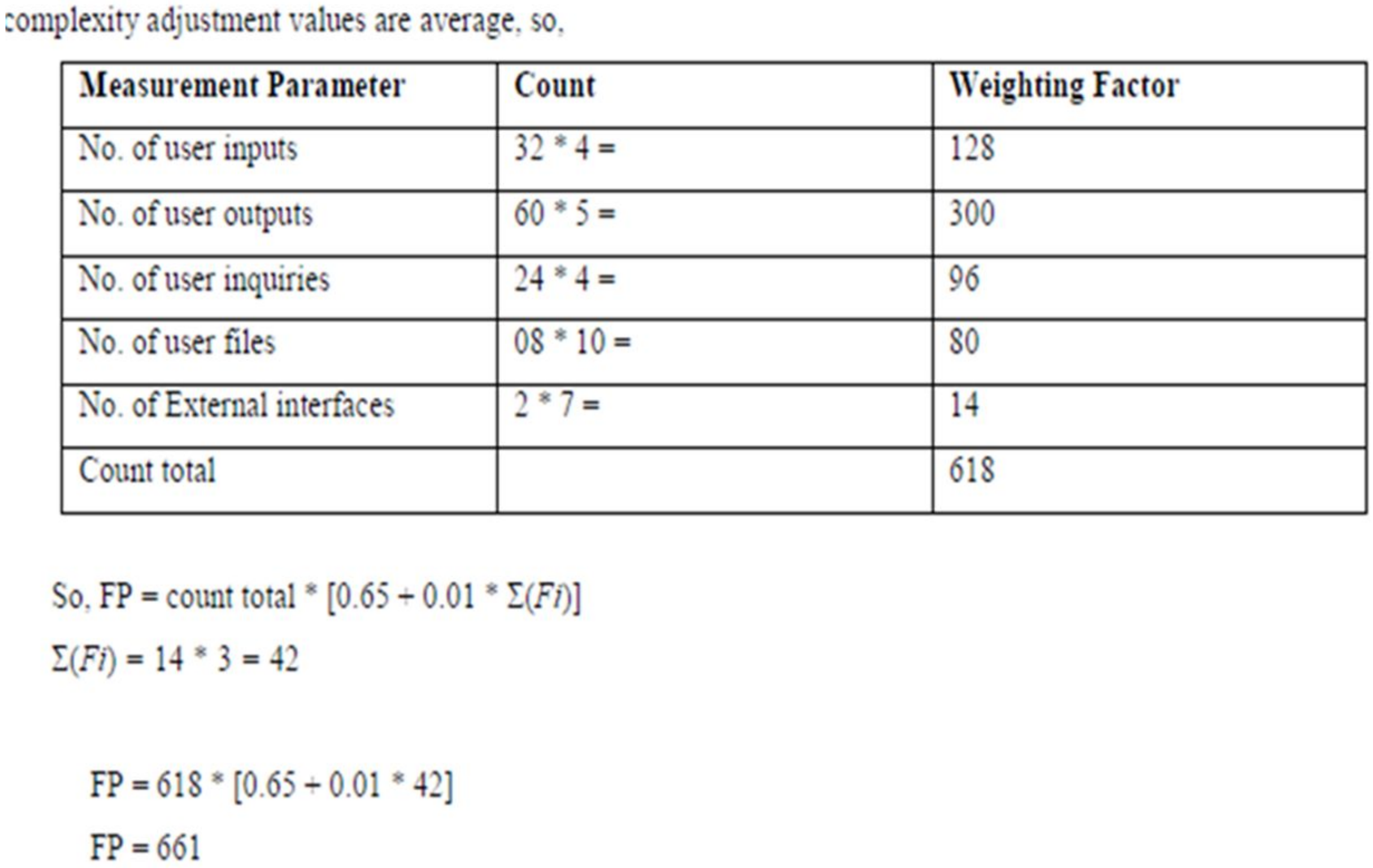
No. of user inquiries = 24

No. of user files = 08

No. of External interfaces = 2

Assume that all complexity adjustment values are average.

FPA : Solution



complexity adjustment values are average, so,

Measurement Parameter	Count	Weighting Factor
No. of user inputs	32 * 4 =	128
No. of user outputs	60 * 5 =	300
No. of user inquiries	24 * 4 =	96
No. of user files	08 * 10 =	80
No. of External interfaces	2 * 7 =	14
Count total		618

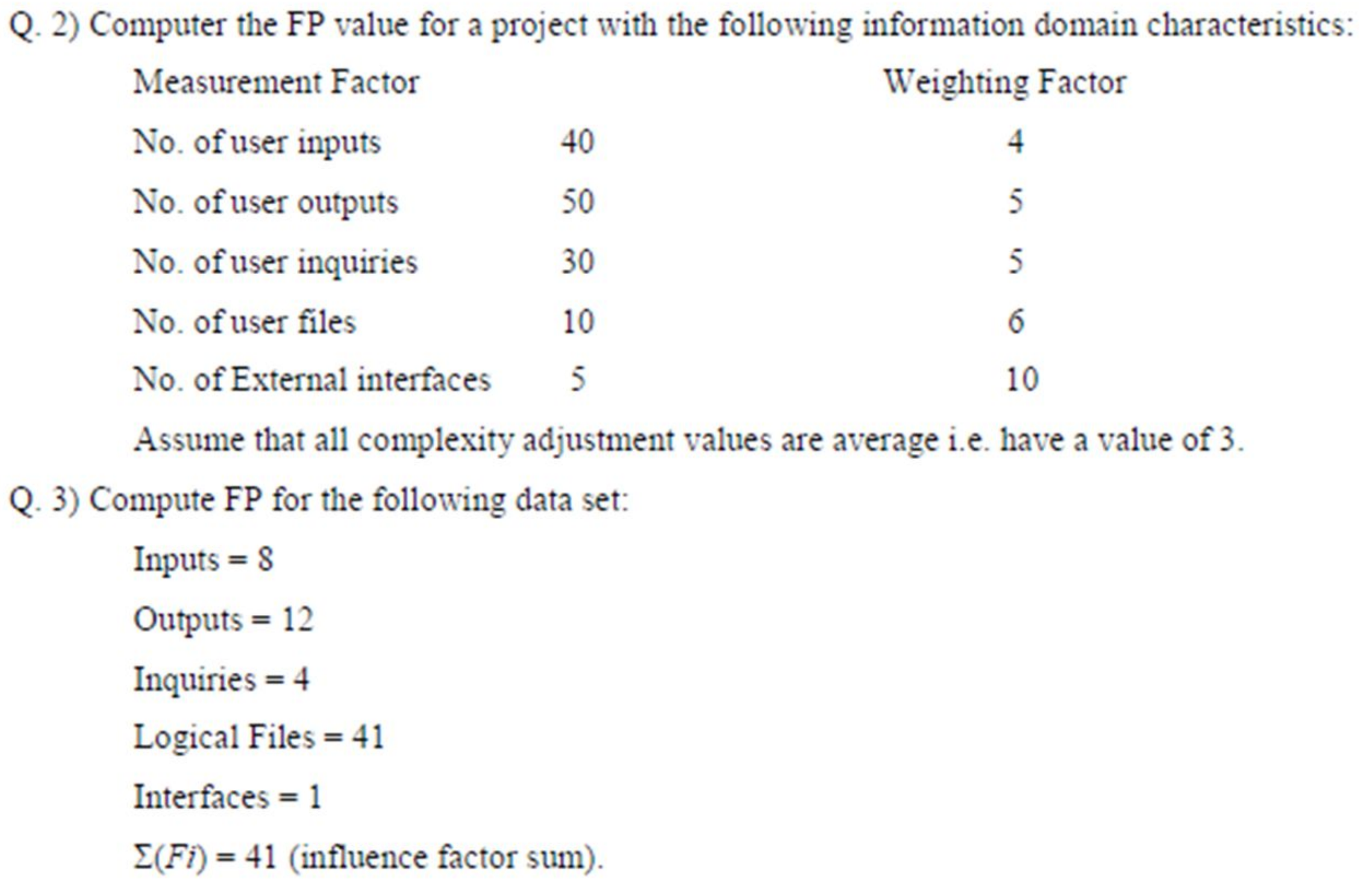
So, FP = count total * [0.65 + 0.01 * $\Sigma(Fi)$]

$\Sigma(Fi)$ = 14 * 3 = 42

FP = 618 * [0.65 + 0.01 * 42]

FP = 661

FPA : Exercise



Q. 2) Computer the FP value for a project with the following information domain characteristics:

Measurement Factor		Weighting Factor
No. of user inputs	40	4
No. of user outputs	50	5
No. of user inquiries	30	5
No. of user files	10	6
No. of External interfaces	5	10

Assume that all complexity adjustment values are average i.e. have a value of 3.

Q. 3) Compute FP for the following data set:

Inputs = 8

Outputs = 12

Inquiries = 4

Logical Files = 41

Interfaces = 1

$\Sigma(Fi) = 41$ (influence factor sum).

FPA : Limitations

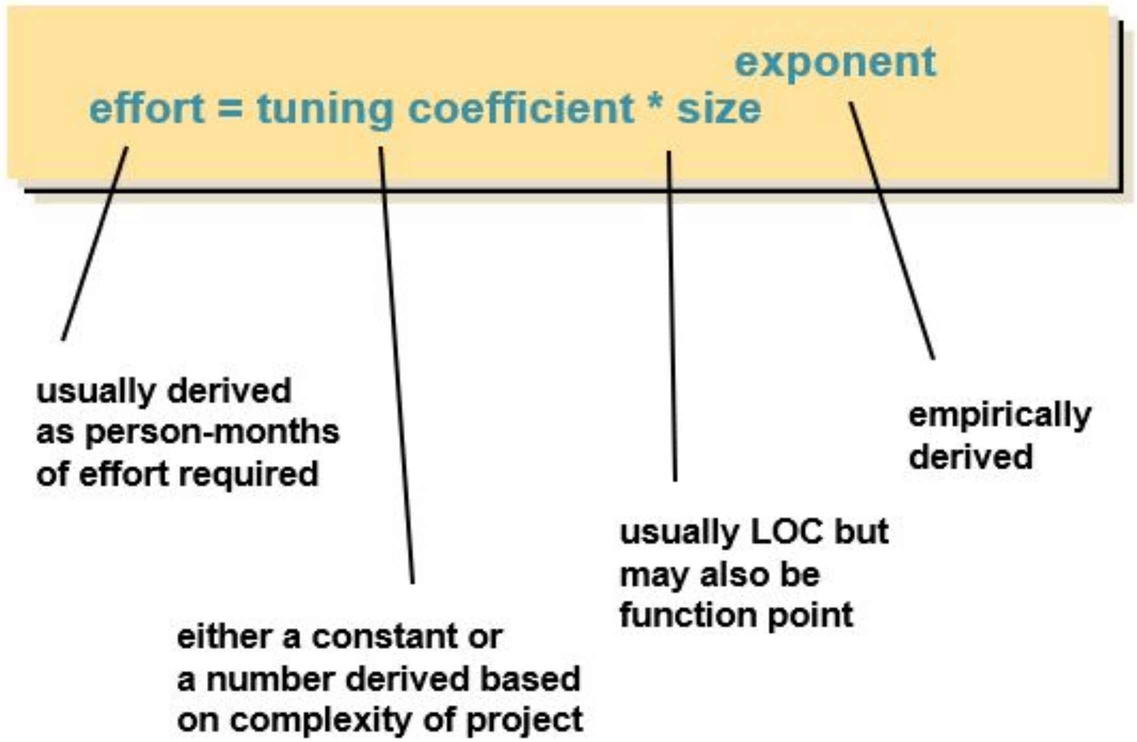
- Function point counts are affected by project size
- Difficult to apply to massively distributed systems or to systems with very complex internal processing
- Difficult to define logical files from physical files
- The validity of the weights that Albrecht established – and the consistency of their application – has been challenged
- Different companies will calculate function points slightly different, making intercompany comparisons questionable

The COCOMO (COnstructive COst MOdel)

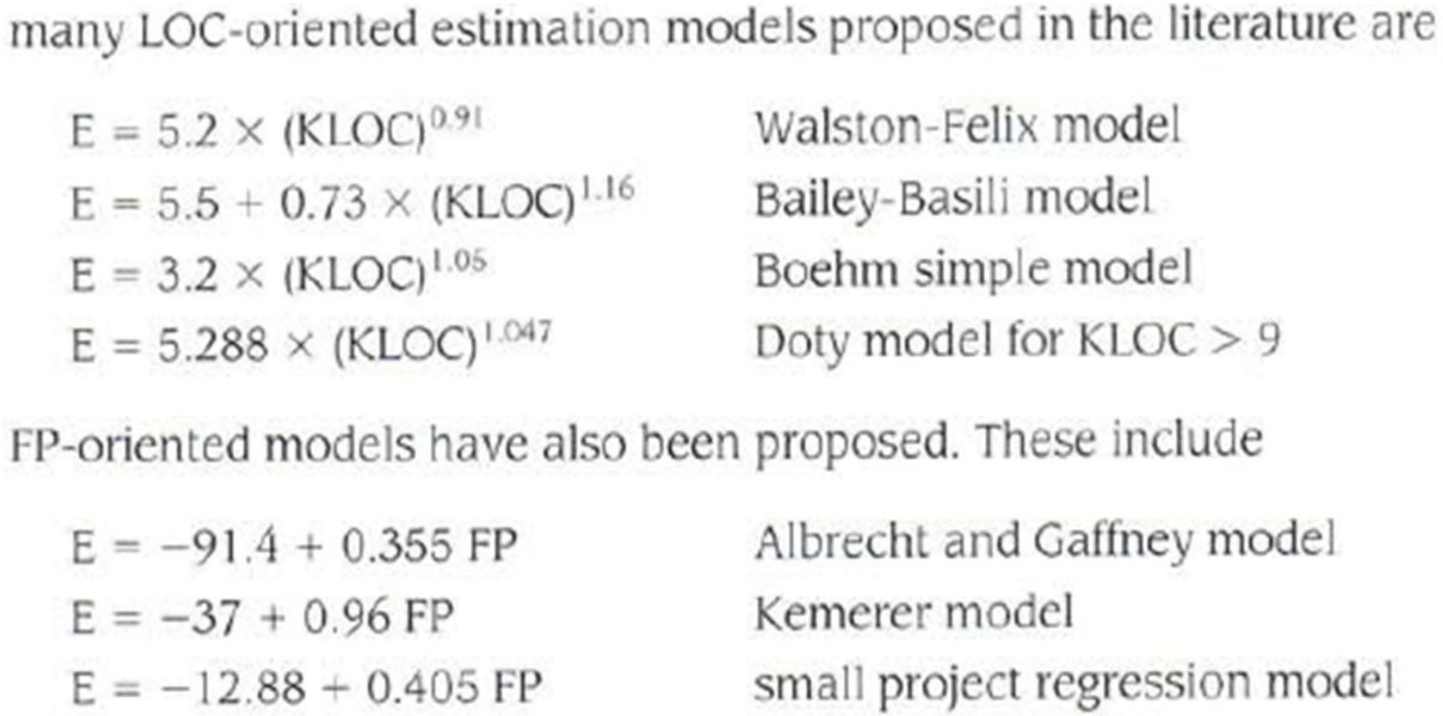
- An empirical model based on project experience
- Well-documented, ‘independent’ model which is not tied to a specific software vendor
- Long history from initial version published in 1981 (COCOMO-81) through various instantiations to COCOMO 2
- **COCOMO 2** takes into account different approaches to **software development, reuse**, etc

The COCOMO (COnstructive COst MOdel)

General form:


exponent
effort = tuning coefficient * size
usually derived as person-months of effort required
either a constant or a number derived based on complexity of project
usually LOC but may also be function point
empirically derived

Estimation Models



many LOC-oriented estimation models proposed in the literature are

$E = 5.2 \times (\text{KLOC})^{0.91}$	Walston-Felix model
$E = 5.5 + 0.73 \times (\text{KLOC})^{1.16}$	Bailey-Basili model
$E = 3.2 \times (\text{KLOC})^{1.05}$	Boehm simple model
$E = 5.288 \times (\text{KLOC})^{1.047}$	Doty model for KLOC > 9

FP-oriented models have also been proposed. These include

$E = -91.4 + 0.355\ \text{FP}$	Albrecht and Gaffney model
$E = -37 + 0.96\ \text{FP}$	Kemerer model
$E = -12.88 + 0.405\ \text{FP}$	small project regression model

COCOMO : Development Modes

- **Organic Mode:** The project is developed in a familiar, stable environment, and the product is similar to previously developed products. The product is relatively small, and requires little innovation.
- **Semi Detached Mode:** The project is developed by experienced and inexperienced staff. The project's characteristics are intermediate between Organic and Embedded.
- **Embedded Mode:** The project is characterized by tight, inflexible constraints and interface requirements. Experience level is low. So, an embedded mode project will require a great deal of innovation.

COCOMO II Model

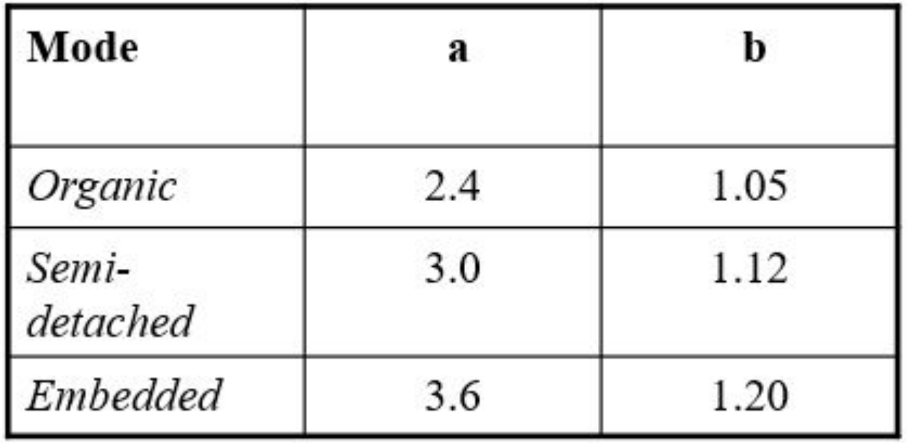
- **Application composition model** - Used during the early stages of software engineering when the following are important
  - Prototyping of user interfaces
  - Consideration of software and system interaction
  - Assessment of performance
  - Evaluation of technology maturity
- **Early design stage model** – Used once requirements have been stabilized and basic software architecture has been established
- **Post-architecture stage model** – Used during the construction of the software

COCOMO : Effort Computation

- The **Basic COCOMO model** computes effort as a function of program size. The Basic COCOMO equation is:

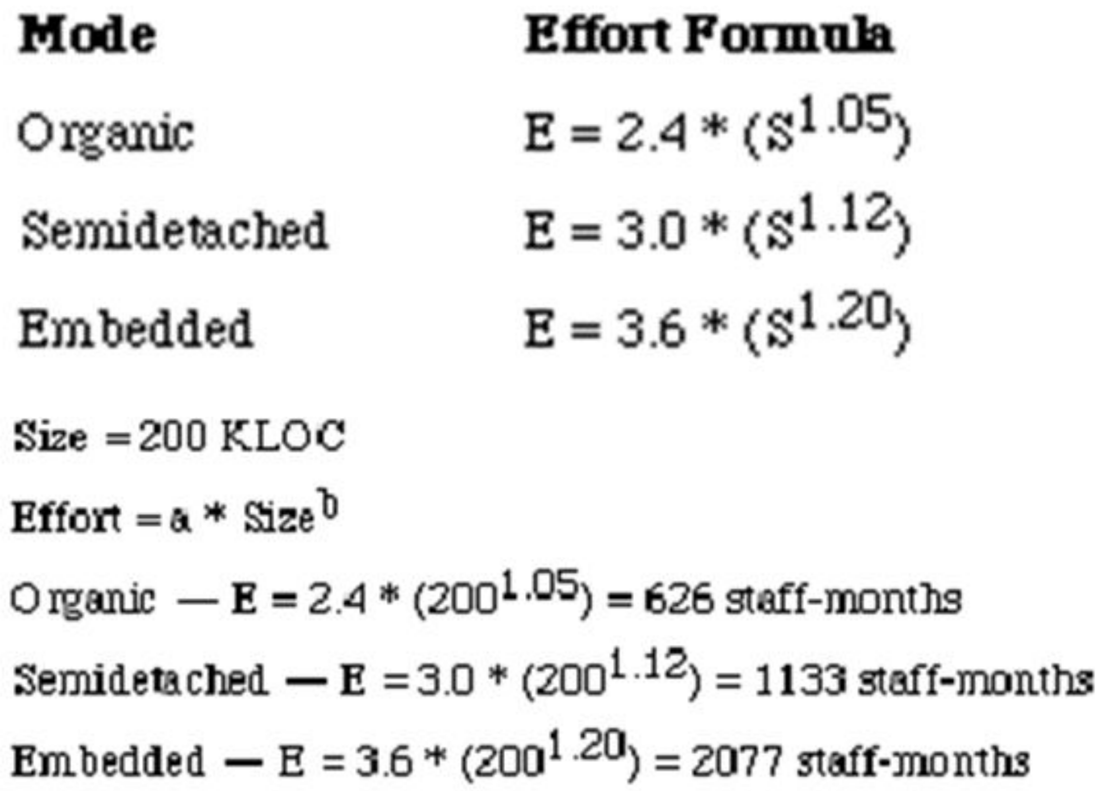
$$E = a(KLOC)^b$$

- Effort for three modes of Basic COCOMO.



Mode	**a**	**b**
Organic	2.4	1.05
Semi-detached	3.0	1.12
Embedded	3.6	1.20

COCOMO : Effort Computation



Mode	**Effort Formula**
Organic	$E = 2.4 * (S^{1.05})$
Semidetached	$E = 3.0 * (S^{1.12})$
Embedded	$E = 3.6 * (S^{1.20})$

Size $= 200$ KLOC

Effort $= a * \text{Size}^{b}$

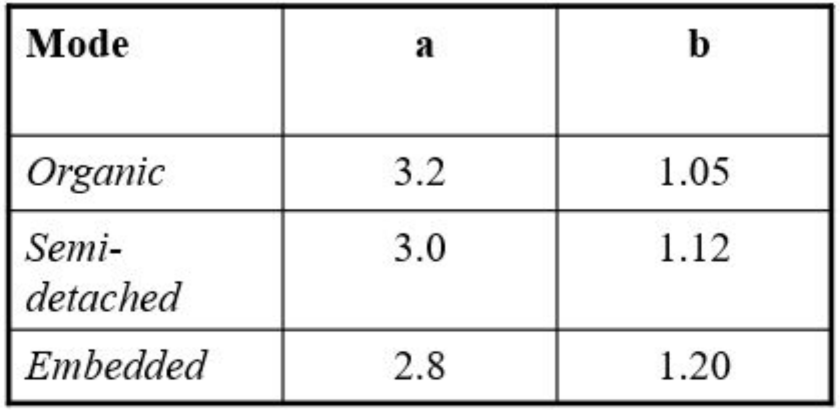
Organic — $\mathbf{E} = 2.4 * (200^{1.05}) = 626$ staff-months

Semidetached — $\mathbf{E} = 3.0 * (200^{1.12}) = 1133$ staff-months

Embedded — $E = 3.6 * (200^{1.20}) = 2077$ staff-months

COCOMO : Effort Computation

- The **intermediate COCOMO model** computes effort as a function of program size and a set of cost drivers. The Intermediate COCOMO equation is:
- $E = a(KLOC)^{b} * EAF$
- Effort for three modes of intermediate COCOMO



Mode	**a**	**b**
Organic	3.2	1.05
Semi-detached	3.0	1.12
Embedded	2.8	1.20

COCOMO : Effort Adjustment Factor

Cost Driver	Very Low	Low	Nominal	High	Very High	Extra High
Required Reliability	.75	.88	1.00	1.15	1.40	1.40
Database Size	.94	.94	1.00	1.08	1.16	1.16
Product Complexity	.70	.85	1.00	1.15	1.30	1.65
Execution Time Constraint	1.00	1.00	1.00	1.11	1.30	1.66
Main Storage Constraint	1.00	1.00	1.00	1.06	1.21	1.56
Virtual Machine Volatility	.87	.87	1.00	1.15	1.30	1.30
Comp Turn Around Time	.87	.87	1.00	1.07	1.15	1.15
Analyst Capability	1.46	1.19	1.00	.86	.71	.71
Application Experience	1.29	1.13	1.00	.91	.82	.82
Programmers Capability	1.42	1.17	1.00	.86	.70	.70
Virtual machine Experience	1.21	1.10	1.00	.90	.90	.90
Language Experience	1.14	1.07	1.00	.95	.95	.95
Modern Prog Practices	1.24	1.10	1.00	.91	.82	.82
SW Tools	1.24	1.10	1.00	.91	.83	.83
Required Dev Schedule	1.23	1.08	1.00	1.04	1.10	1,10

COCOMO : Solution

	Organic	Semidetached	Embedded
a	3.2	3.0	2.8
b	1.05	1.12	1.20

Mode	Effort Formula
Organic	$E = 3.2 * (S^{1.05}) * C$
Semidetached	$E = 3.0 * (S^{1.12}) * C$
Embedded	$E = 2.8 * (S^{1.20}) * C$

e.g. Size = 200 KLOC

$\text{Effort} = a * \text{Size}^b * C$

Cost drivers:

Low reliability	.88
High product complexity	1.15
Low application experience	1.13
High programming language experience	.95

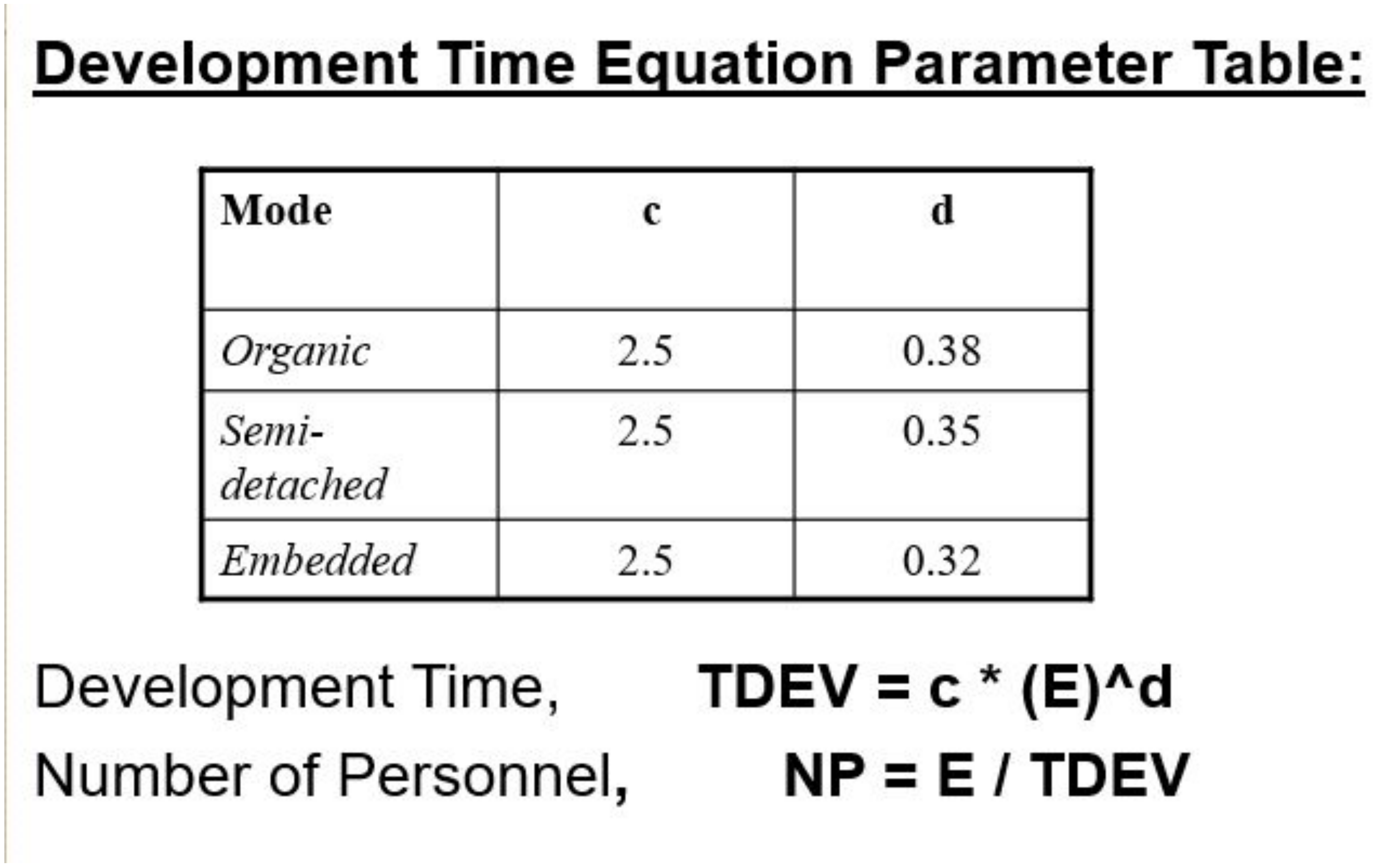
$C = .88 * 1.15 * 1.13 * .95 = 1.086$

Organic — $E = 3.2 * (200^{1.05}) * 1.086 = 906$ staff-months

Semidetached — $E = 3.0 * (200^{1.12}) * 1.086 = 1231$ staff-months

Embedded — $E = 2.8 * (200^{1.20}) * 1.086 = 1755$ staff-months

COCOMO : Solution



Development Time Equation Parameter Table:

Mode	**c**	**d**
Organic	2.5	0.38
Semi-detached	2.5	0.35
Embedded	2.5	0.32

Development Time, **TDEV = c * (E)^d**

Number of Personnel, **NP = E / TDEV**

Basic COCOMO : Exercise

- Consider one module of student information system is to be developed. LOC is estimated to be 8,000. Compute how many programmer months and programmers will be required using Basic COCOMO model.

Basic COCOMO : Solution

- $E = 2.4 (8)^{1.05}$

  $= 18$ programmer month

- $TDEV = 2.5 (18)^{0.38}$

  $= 6$ months

- $NP = E/TDEV$

  $= 18 / 6 = 3$ programmers

Intermediate COCOMO : Exercise

- System for office automation must be designed. Four modules in system are data entry, data update, query and report generator. Project fall in organic category. Sizes for modules are as:
  - Data entry:0.6 KDSI
  - Data update:0.6 KDSI
  - Query:0.8 KDSI
  - Report generator: 1.00 KDSI.
- Assume all factors had nominal rating. Find out overall effort estimate

Intermediate COCOMO : Solution

Total KDSI= 0.6+0.6+0.8+1.0=3 KDSI

EAF = product of all factors i.e. 1.00

Ei = a(KDSI)^b

= 3.2(3)^1.05

= 10.14 PM

E = Ei * EAF

= 10.14 * 1.00

= 10.14 PM