ESTIMATION FOR SOFTWARE PROJECTS

Unit II

Text Book: Software Engineering

- By Roger Pressman – 8th ed.

Topics

- Project planning process
- Resources
- Decomposition techniques
- Empirical Estimation Models
- Estimation of OO projects

Introduction

oftware project management begins with a set of activities at are collectively called ***project planning***

efore project begin, the software team should estimate

✓ The work to be done

✓Resources

✓The time that will elapse from start to finish

✓Establishing the schedule

✓Defining the tasks and milestones

✓Specifying the inner task specifications

Observation on Estimation

- Estimation of resources, cost, and schedule for a software engineering effort requires
  - Experience
  - Access to good historical information (metrics)
  - The courage to commit to quantitative predictions when qualitative information is all that exists
  - Carries inherent risk and this risk leads to uncertainty.

Project Planning Objectives

To provide a framework that enables the manager to mak easonable estimates for.

- Resources
- Cost and
- Schedule

Estimates should attempt to define best-case and worst-cas cenarios so that project outcomes are bounded.

The project plan must be adapted and updated as the projec proceeds.

Factors that effect the estimates

.Project complexity

. Project size

. Degree of structural uncertainty

oject complexity: Complexity is a relative measure that is affected by miliarity with past effort.

oject size: As size increases, the

terdependency among various elements of the software grows rapidly. oblem decomposition is the solution

egree of structural uncertainty:

ructure refers to the degree to which requirements have been solidified e ease with which functions can be compartmentalized, and the erarchical nature of the information that must be processed

Task Set for Project Planning

Establish Project Scope

Determine Feasibility

Analyze Risks

Define required resources

- Determine required human resources.
- Define usable software resources.
- Identify environmental resources.

Estimate cost and effort.

- Decompose the problem.
- Develop two (or) more estimates using size, function points, process tasks (or) use cases.
- Reconcile(merge) the estimates.

Develop a project schedule

- Establish the meaningful task set.
- Define a task network.
- Use scheduling tools to develop a time-line chart.
- Define schedule tracking mechanisms.

Steps of Estimation

Begins with a description of scope of the product.

Software scope describes the functions and features that are to be delivered to end users; the data that are input and output; the "content" that is presented to users as a consequence of using the software; and the performance, constraints, interfaces, and reliability that bound the system. Scope is defined using one of two techniques:

1. A narrative description of software scope is developed afte communication with all stakeholders.

2. A set of use cases is developed by end users.

The problem is decomposed into set of smaller problems.

And each of these estimated with historical data and experiences.

Problem complexity and the risks are considered before fina estimation

Resources

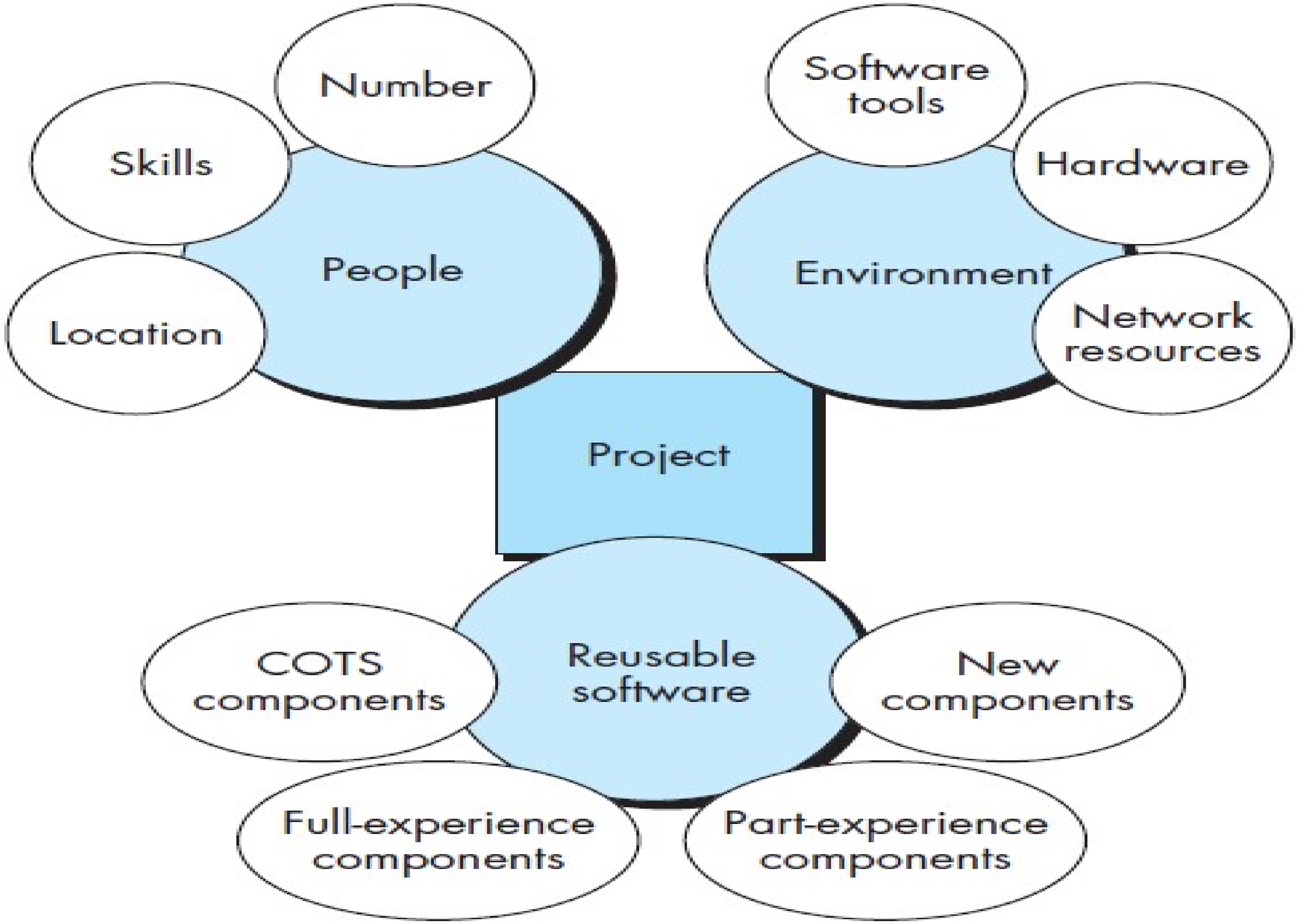
Estimation of resources required to accomplish the software development effort.

- Human Resources
- Reusable Software
- Environmental Resources (Both Software and Hardware)

Each resource is specified with **4 characteristics**,

1. Description of the resource
2. Statement of availability
3. Time when the resource will be required
4. Duration of time that the resource will be applied.

Resources


Number
Skills
People
Location
Software tools
Hardware
Environment
Network resources
Project
Reusable software
COTS components
New components
Full-experience components
Part-experience components

Human Resources

he **Planner** begins by **evaluating software scope** and **selecting the ski equired to complete development**.

rganizational Position

- manager, senior software engineer

pecialty

- telecommunications, database, client-server

mall Projects

- A single individual may perform all software engineering tasks.

arger Projects

- The software team may be geographically dispersed across a number of differe locations.

Number of people required for a project can be determined only after an estimate evelopment effort (Person-months)

Environmental Resources

✓The environment that supports a software project, often called th software engineering environment (SEE), incorporates hardware ar software.

✓ Hardware

✓ Software

Reusable Software Requirements

omponent-based software engineering (CBSE) emphasizes reusability—that is, the creation and reuse of software building blocks.

Such building blocks, often called *components,* must be catalogued for easy reference, standardized for easy application, and validated for easy integration.

There are four software resource categories:

- *Off-the-shelf components*
- *Full-experience components*
- *Partial-experience components*
- *New components*

Reusable Software Requirements

off-the-shelf components (existing software that can be acquired from a third party or from a past project),

full-experience components (existing specifications, designs, code, or test data developed for past projects that are similar to the software to be built for the current project),

partial-experience components (existing specifications, designs, code, or test data developed for past projects that are related to the software to be built for the current project but will require substantial modification),

new components (components built by the software team specifically for the needs of the current project).

Project Estimation Options

Four Options for achieving reliable cost and effort estimates

1) Delay estimation until late in the project (we should be able to achieve 100 accurate estimates after the project is complete)
2) Base estimates on similar projects that have already been completed
3) Use relatively simple decomposition techniques to generate project cost and effort estimates
4) Use one or more empirical estimation models for software cost and effort estimation

Option #1 is not practical, but results in good numbers

Option #2 can work reasonably well, but it also relies on other project influences being roughly equivalent

Options #3 and #4 can be done in tandem(cycle) to cross check each other

Project Estimation

- A model is based on experience (historical data) and takes the form:

$$d = f(v_i)$$

- *d = one of a number of estimated values (effort, cost,* project duration)
- *vi = selected independent parameters*

Project Estimation Approaches

Decomposition techniques(LOC and FP Based)

- ✓ These take a "divide and conquer" approach
- ✓ Cost and effort estimation are performed in a stepwise fashion by breaking down a project into major functions and related software engineering activities

Empirical estimation models (COCOMO)

- ✓ Can be used to complement(supplement) decomposition techniques
- ✓ Offer a potentially valuable estimation approach if the historical dat used to seed the estimate is good

Decomposition Techniques

- Before an estimate can be made and decomposition techniques (**Divide and conquer decomposition, big-bang unUniform modules – Top down** & **Integral-Bottom up composition**) applied, the planner must
  - Understand the scope of the software to be built
  - Generate an estimate of the software's size
- **Then two approaches are used**
  - Problem-based estimation
    - Based on either source lines of code or function point estimates
  - Process-based estimation
    - Based on the effort required to accomplish each task

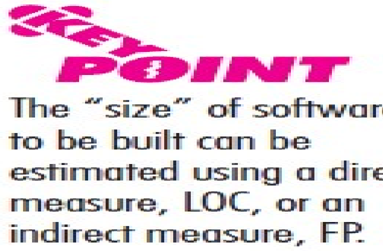
Software Sizing

Accuracy of a software project estimate is predicated on a number of things

- the degree to which you have **properly estimated the size of the product** to be built
- the ability to **translate the size estimate into human effort, calendar time, and dollars**
- the degree to which the **project plan reflects the abilities of the software team**
- the **stability of product requirements and the environment** that supports the **Software Engineering Effort (SEE)**.

Software Sizing

- Project estimate is only as good as the estimate of the size of work to be accomplished.
- Sizing represents the first major challenge as a planner.



KEY POINT

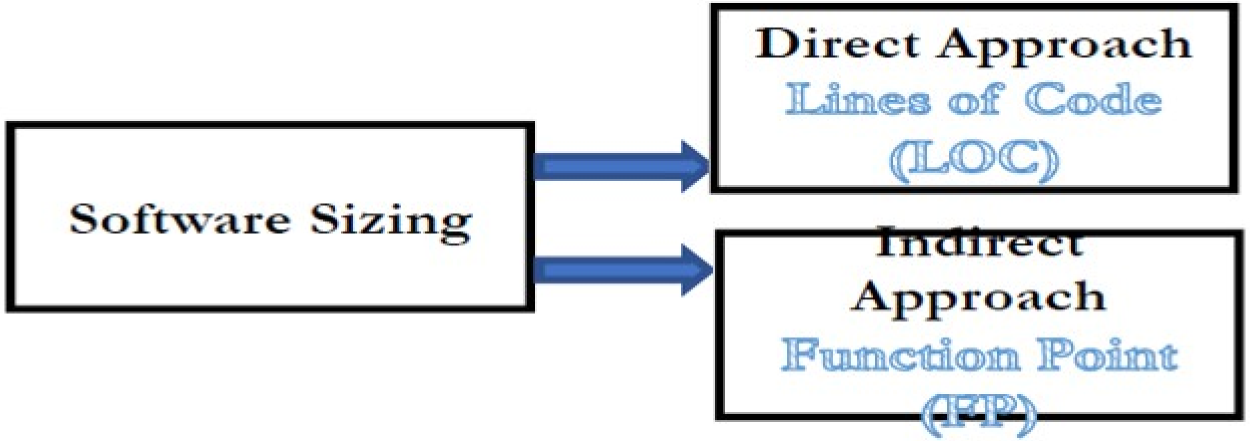
The "size" of softwar to be built can be estimated using a dire measure, LOC, or an indirect measure, FP.

Software Sizing

ize refers to a **quantifiable outcome** of the software project.

f a **direct approach is taken**, size can be measured in **lines of code (LOC)**.

f an **indirect approach is chosen**, size is represented as **function points (FP)**.



Approaches to Software Sizing

utnam and Myers [Put92] suggest **four different approaches** to the **sizing problem**.

unction point sizing

- The **planner develops estimates** of the **information domain characteristics**.

tandard Component Sizing

- Composed of a number of different "**standard components**" that are generic to a parti application area.
- **Standard components** includes,
  - modules
  - screens
  - reports
  - interactive programs
  - batch programs
  - files
  - LOC

– Estimate the number of occurrences of each standard component

– Use historical project data to determine the delivered LOC size per standard componen

Software Sizing

Fuzzy logic sizing

- uses the approximate reasoning techniques that are the cornerstone of fuz logic.
- To apply this approach,
  - planner must identify the type of application.
  - establish its magnitude on a qualitative scale.
  - then refine the magnitude within the original range

Change sizing

- – Used when changes are being made to existing software
- – Estimate the number and type of modifications that must be accomplished
- – Types of modifications include reuse, adding code, changing code, and deleting code
- – An effort ratio is then used to estimate each type of change and the size of the change

Putnam and Myers suggest that results of each of these sizing approaches combin statistically to **create three-point** (or) **expected value estimate.**

Decomposition techniques

- LOC – Size oriented metric – KLOC (Thousand lines of code)
- FP – Function oriented metric
  - *Computation of the function point is based on characteristics of the software's information domain and complexity*

FP metric can be used to

(1) estimate the cost or effort required to design, code, and test the software

(2) predict the number of errors that will be encountered during testing

(3) forecast the number of components and/or the number of projected source lines in the implemented system.

Software Effort Estimation

- What is meant by software effort estimation?
- In software development, effort estimation is the process of predicting the most realistic amount of effort (expressed in terms of person-hours or money) required to develop or maintain software based on incomplete, uncertain and noisy input.

Problem based Estimation

ines Of Code (LOC) and Function Points (FP) are used in two ways during softwa roject estimation.

- An estimation variable to size each element of the software.
- A baseline metrics collected from past projects and used in conjunction wi estimation variables to develop cost and effort projections.
- LOC and FP estimation are distinct estimation techniques but have a commo number of characteristics.
- Baseline productivity metrics (LOC/pm or FP/pm) are then applied to th appropriate estimation variable.
- The LOC and FP estimation techniques differ in the level of detail required f decomposition and the target of the partitioning. When LOC is used as th estimation variable, decomposition is absolutely essential and is often taken considerable levels of detail. The greater the degree of partitioning, the mo likely reasonably accurate estimates of LOC can be developed.

Problem based Estimation

For FP estimates, decomposition works differently. Rather tha ocusing on function, each of the information domain characteristics- nputs, outputs, data files, inquiries, and external interfaces—as well a he 14 complexity adjustment values are estimated. The resultar stimates can then be used to derive an FP value that can be tied to pa ata and used to generate an estimate.

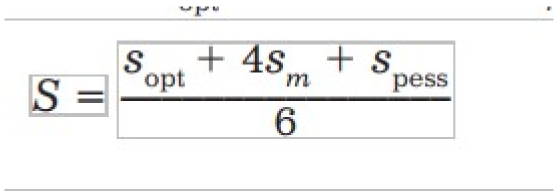
Regardless of the estimation variable that is used, the estimating begir vith a range of values for each function or information domain valu sing historical data estimate an optimistic, most likely, ar essimistic size value for each function or count for each informatio omain value. An implicit indication of the degree of uncertainty rovided when a range of values is specified.

Problem based Estimation

A three-point or expected value can then be computed. The expecte alue for the estimation variable (size) S can be computed as a weighte verage of the optimistic (sopt), most likely (sm), and pessimistic (spes stimates. For example,

gives the “most likely” estimate.

Once the expected value for the estimation variable has bee etermined, historical LOC or FP productivity data are applied.



$$S = \frac{s_{\text{opt}} + 4s_m + s_{\text{pess}}}{6}$$

Example of LOC-Based Estimation

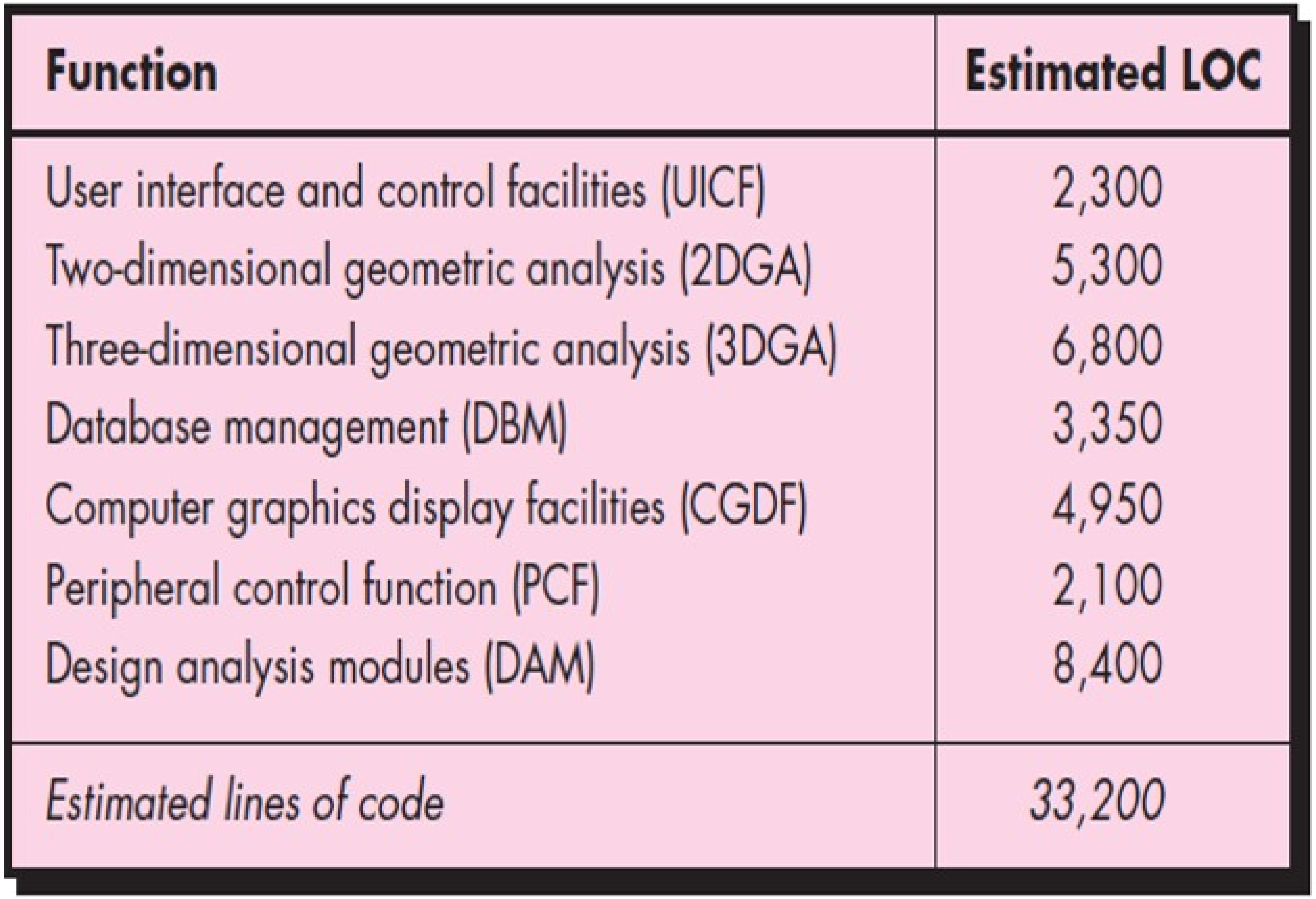
Software package to be developed for Computer-Aided Des Application

A preliminary statement of software scope can be developed.

Before estimation can begin, the planner must determine wh characteristics of good human/machine interface design” means or wh the size and sophistication of the “**CAD database**” are to be.

A range of LOC estimates is developed for each function.

LOC based Estimation



Function	Estimated LOC
User interface and control facilities (UICF)	2,300
Two-dimensional geometric analysis (2DGA)	5,300
Three-dimensional geometric analysis (3DGA)	6,800
Database management (DBM)	3,350
Computer graphics display facilities (CGDF)	4,950
Peripheral control function (PCF)	2,100
Design analysis modules (DAM)	8,400
Estimated lines of code	*33,200*

LOC based Estimation

For example, the range of LOC estimates for the 3D geometric analysis function is optimistic, 4600 LOC; most likely, 6900 LOC and pessimistic, 8600 LOC. Applying Equation (33.1), the expected value for the 3D geometric analysis function is 6800 LOC.

Other estimates are derived in a similar fashion
Estimated value for 3D Geometric Function =
(4600+4*6900+8600)/6=40800/6=6800

LOC based Estimation

A range of LOC estimates is developed for each function. For example, the range LOC estimates for the 3D geometric analysis function is optimistic, 4600 LOC;mo likely, 6900 LOC; and pessimistic, 8600 LOC.

The expected value for the 3D geometric analysis function is 6800 LOC. Oth estimates are derived in a similar fashion. By summing vertically in the estimat LOC column, an estimate of 33,200 lines of code is established for the CAD system

A review of historical data indicates that the organizational average productivi for systems of this type is 620 LOC/pm.

Based on a burdened labor rate of$8,000 per month, the cost per line of code approximately $13(8000/620). Based on the LOC estimate and the historic productivity data, the total estimated project cost is $431,600(33,200*13) ar the estimated effort is 54 person-months.

Solution

y 1 =>

Total Effort = Total LOC/Productivity

33200/620=53.54 ≈ 54 person-months

=> 6 developers

ffort = Total Effort/6 = 54/6 = 9 months

otal Cost = Total Effort * Labor Rate

4 * 8000 ≈ $4,32,000

y2 =>

Cost per LOC = Labor Rate /Productivity

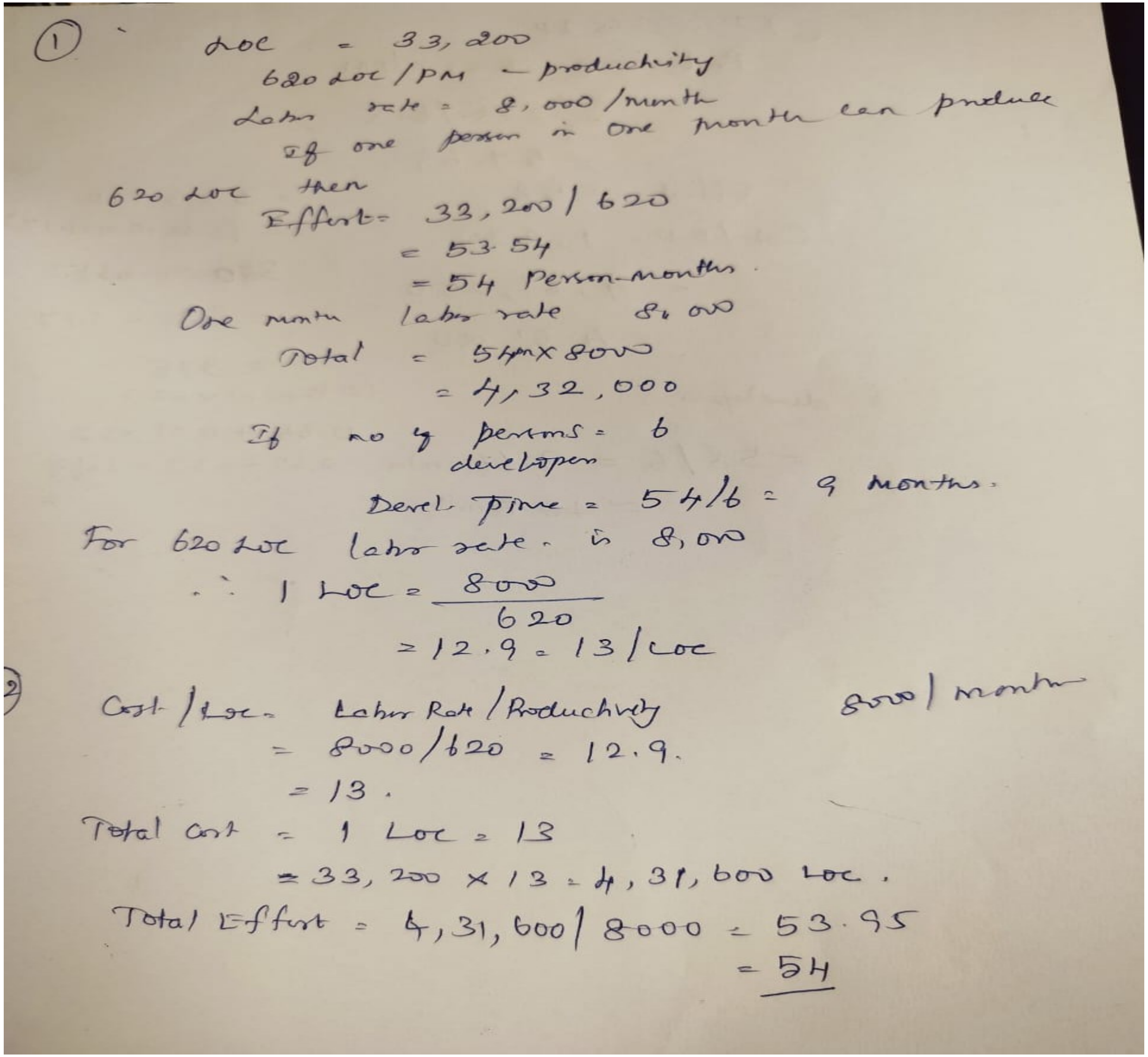
8000/620=$1.29 ≈ $13

=> Total Cost = Total LOC * Cost per LOC

33,200* 13= $431,600

> Total Effort = Total Cost / Labor Rate -> 431,600/8000 = 54 person-months

Solution



① Loc = 33,200

620 Loc / PM — productivity

Labor rate = 8,000 / month

If one person in one month can produce 620 Loc then

Effort = 33,200 / 620

= 53.54

= 54 Person-months.

One month labor rate 8,000

Total = 54m X 8000

= 4,32,000

If no of persons = 6 developers

Devel. Time = 54/6 = 9 months.

For 620 Loc labor rate is 8,000

∴ 1 Loc = $\frac{8000}{620}$

= 12.9 = 13 / Loc

② Cost / Loc = Labor Rate / Productivity   8000 / month

= 8000/620 = 12.9.

= 13.

Total cost = 1 Loc = 13

= 33,200 × 13 = 4,31,600 Loc.

Total Effort = 4,31,600 / 8000 = 53.95

= 54

FP-Based Estimation

ecomposition for FP-based estimation focuses on information domain values rather th oftware functions.

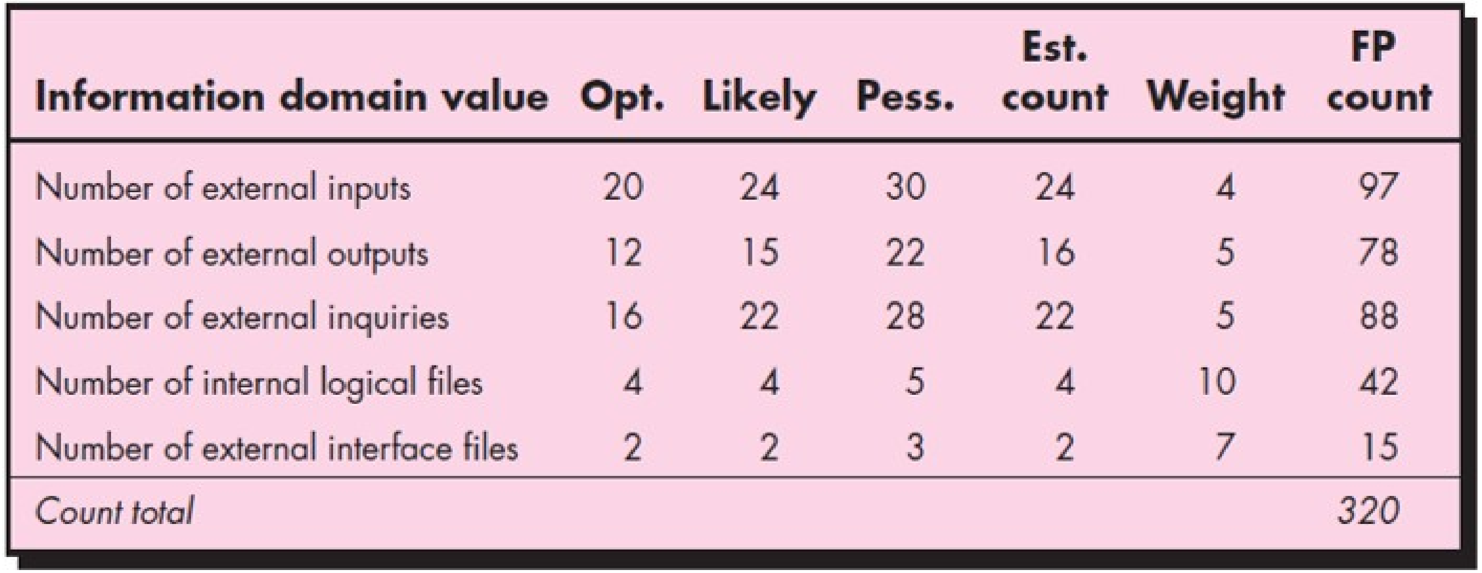
he complexity weighting factor is assumed to be average.

P based estimation for CAD Software

- A **three-point or expected value is computed for each domain characteristic**
  - *expected value for the* estimation variable (size) *S,* weighted average of the optimis (*s_{opt}*), *most likely (*s_m*), and pessimistic (*s_{pess}*) estimates*

$$S = \frac{s_{\mathrm{opt}} + 4s_m + s_{\mathrm{pess}}}{6}$$

Estimating information domain values



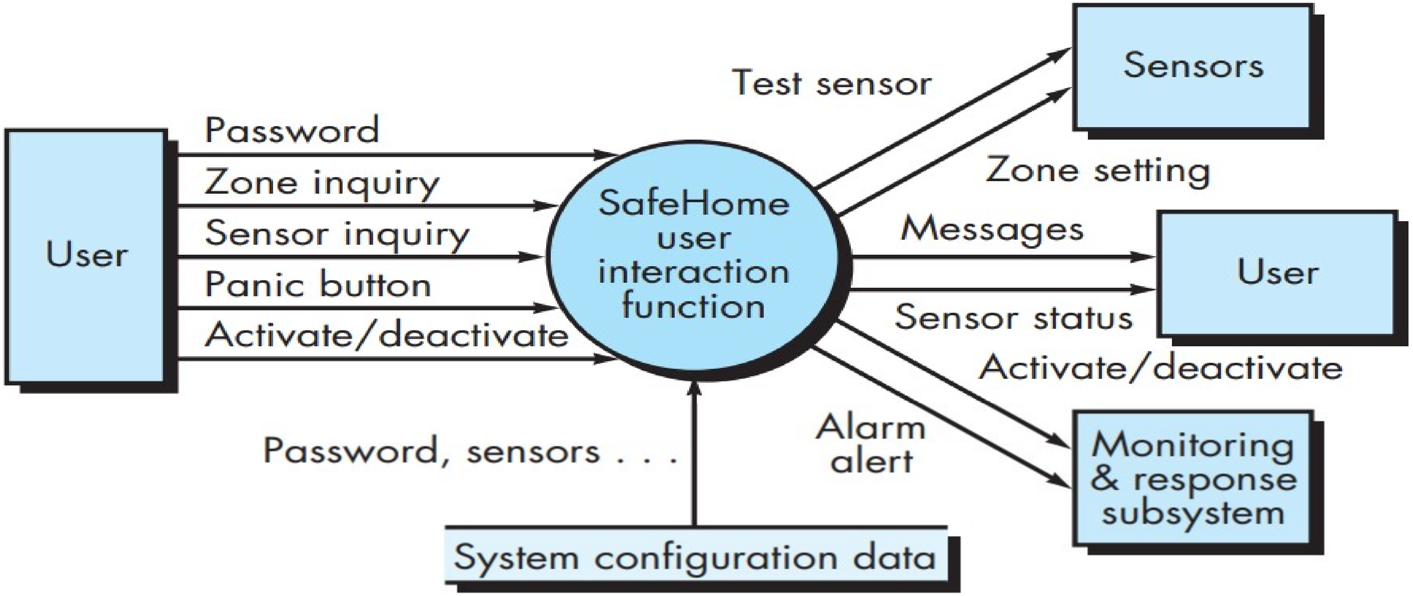
Information domain value	Opt.	Likely	Pess.	Est. count	Weight	FP count
Number of external inputs	20	24	30	24	4	97
Number of external outputs	12	15	22	16	5	78
Number of external inquiries	16	22	28	22	5	88
Number of internal logical files	4	4	5	4	10	42
Number of external interface files	2	2	3	2	7	15
Count total						*320*

FP-Based Estimation

- The function point (FP) metric can be used effectively as a means for measuring the functionality delivered by a system. Using historical data, the FP metric can then be used to
- (1) estimate the cost or effort required to design, code, and test the software;
- (2) predict the number of errors that will be encountered during testing; and
- (3) forecast the number of components and/or the number of projected source lines in the implemented system.

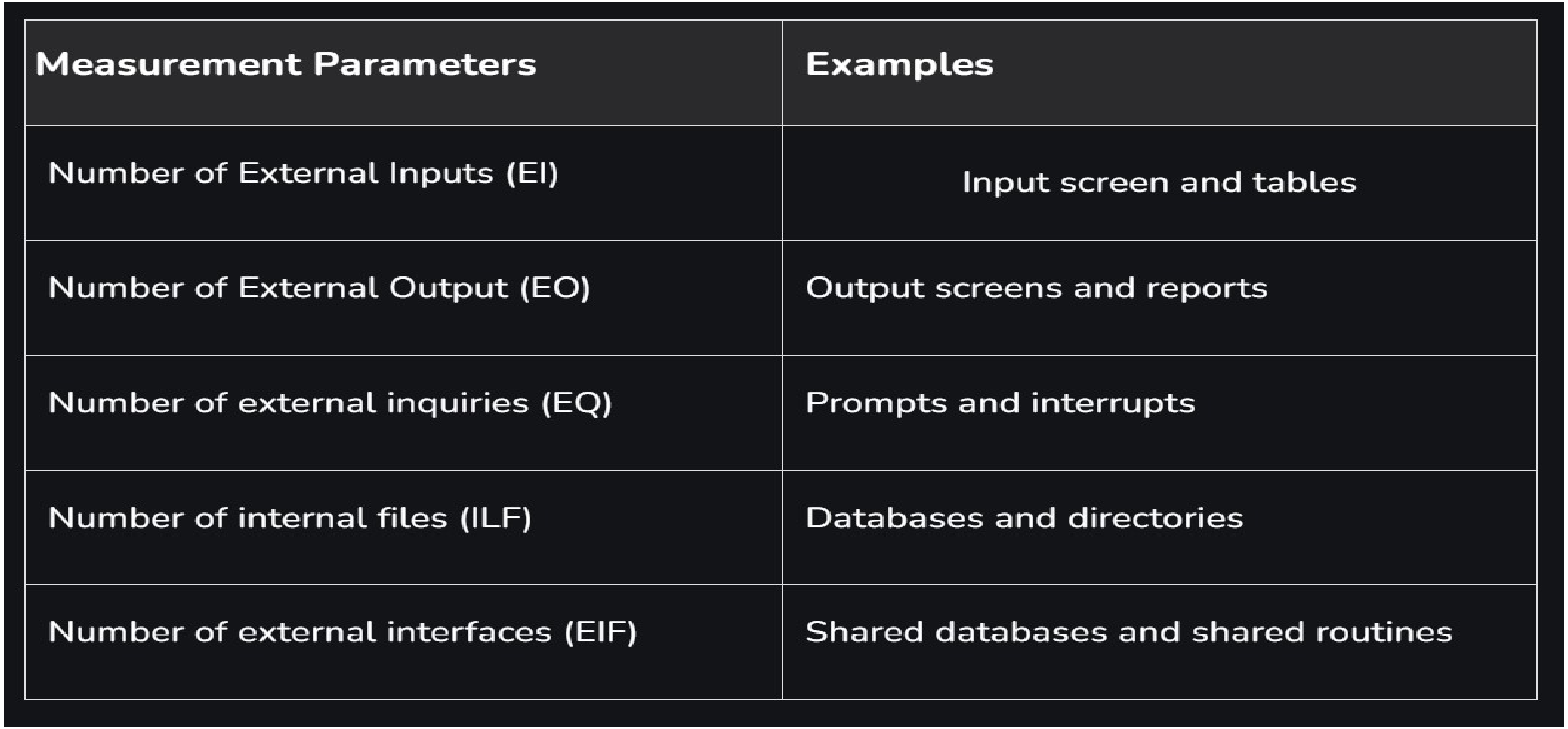
FP-Based Estimation

A flow model
or *SafeHome*
user interac-
ion function



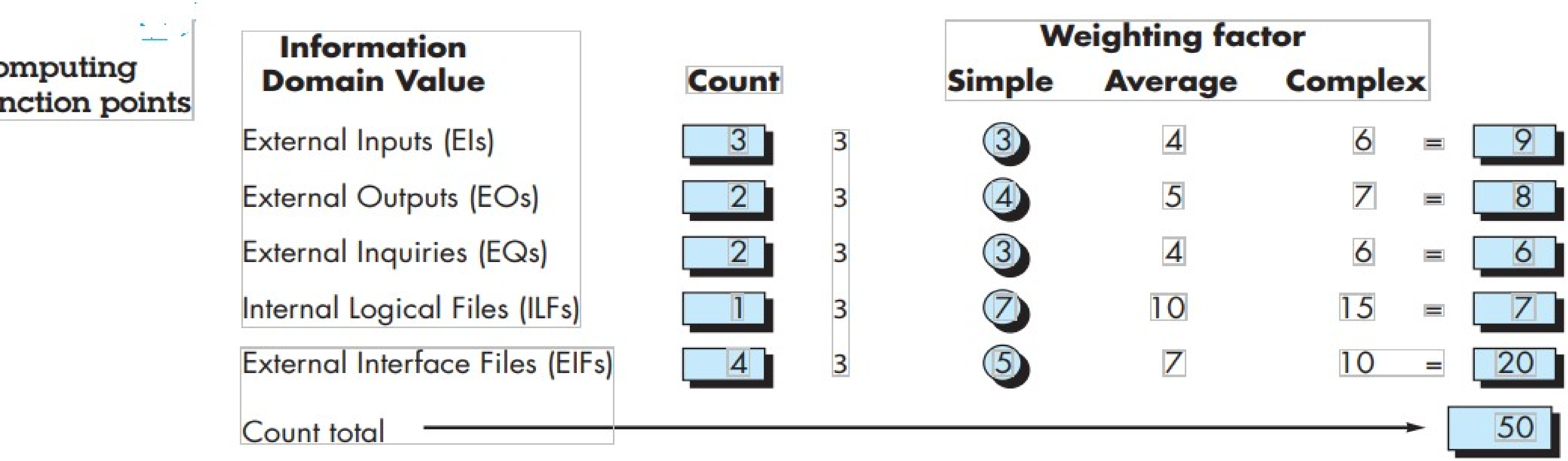
ee external inputs—password, panic button, and activate/deactivate, two external inquiries—zo
iry and sensor inquiry. One ILF (system configuration file) is shown. Two external outputs (messa
sensor status) and four EIFs (test sensor, zone setting, activate/deactivate, and alarm alert)
present.

FP Based Estimation



Measurement Parameters	Examples
Number of External Inputs (EI)	Input screen and tables
Number of External Output (EO)	Output screens and reports
Number of external inquiries (EQ)	Prompts and interrupts
Number of internal files (ILF)	Databases and directories
Number of external interfaces (EIF)	Shared databases and shared routines

FP Based Estimation



omputing
nction points

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

ssumed that S (Fi) is 46 (a moderately complex product). Therefore,

50 * [0.65 1 (0.01 3 46)]= 56

l on the projected FP value derived from the requirements model, the project team can estimate the overal mented size of the SafeHome user inter-action function. Assume that past data indicates that one FP translate 60 lines of code (an object-oriented language is to be used) and that 12 FPs are produced for each person n of effort.

POINT

e adjustment
rs are used to
de an indication
oblem complexity.

The F_i (i = 1 to 14) are *value adjustment factors* (VAF) based on responses to the following questions [Lon02]:

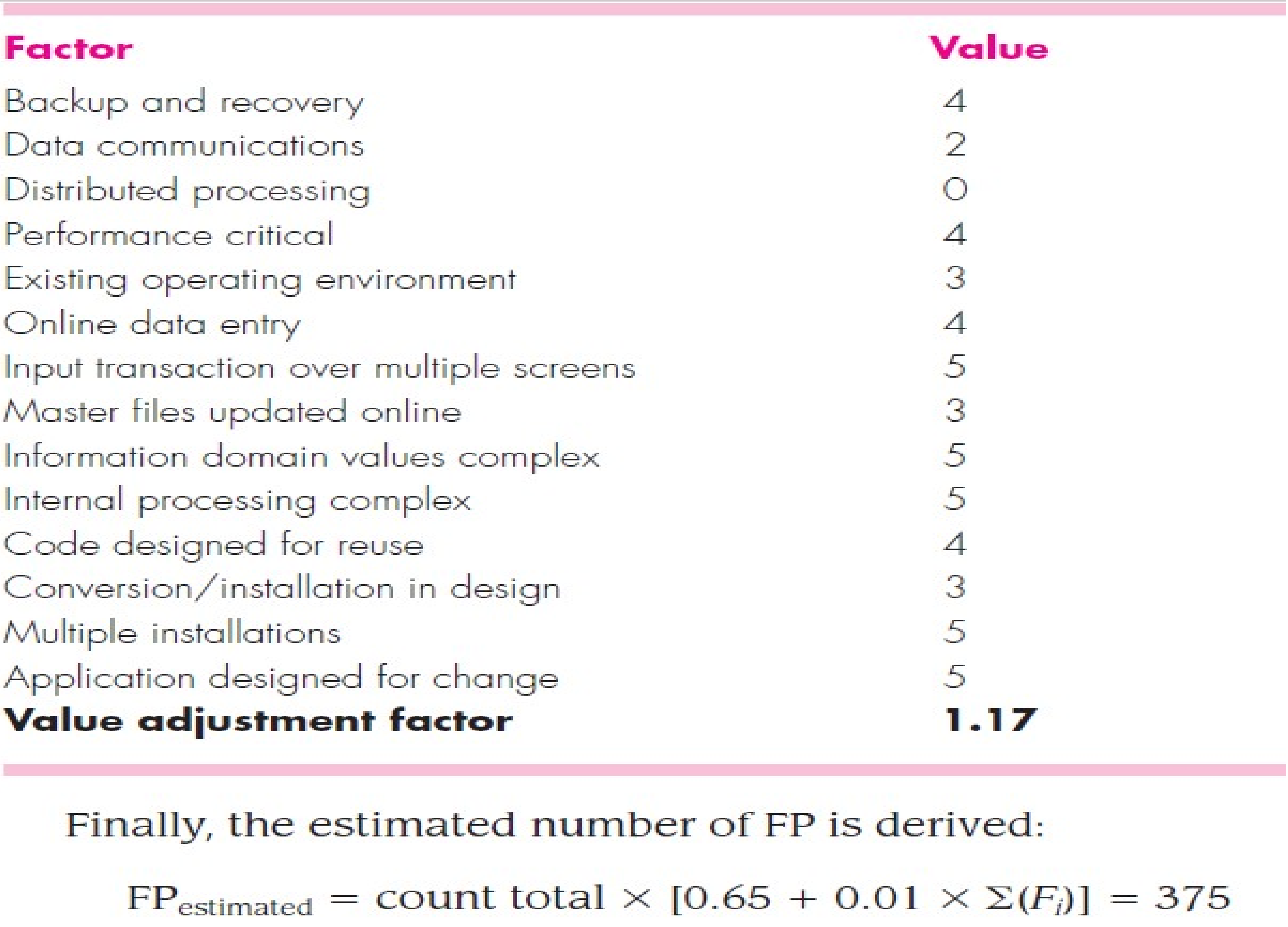
1. Does the system require reliable backup and recovery?
2. Are specialized data communications required to transfer information to or from the application?
3. Are there distributed processing functions?
4. Is performance critical?
5. Will the system run in an existing, heavily utilized operational environment?
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?
8. Are the ILFs updated online?
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 ease of use by the user?

Each of these questions is answered using a scale that ranges from 0 (not important or applicable) to 5 (absolutely essential). The constant values in Equation (23.1) and the weighting factors that are applied to information domain counts are determined empirically.

VebRef

nline FP calculator
be found at

Value/Complexity Adjustment Factor

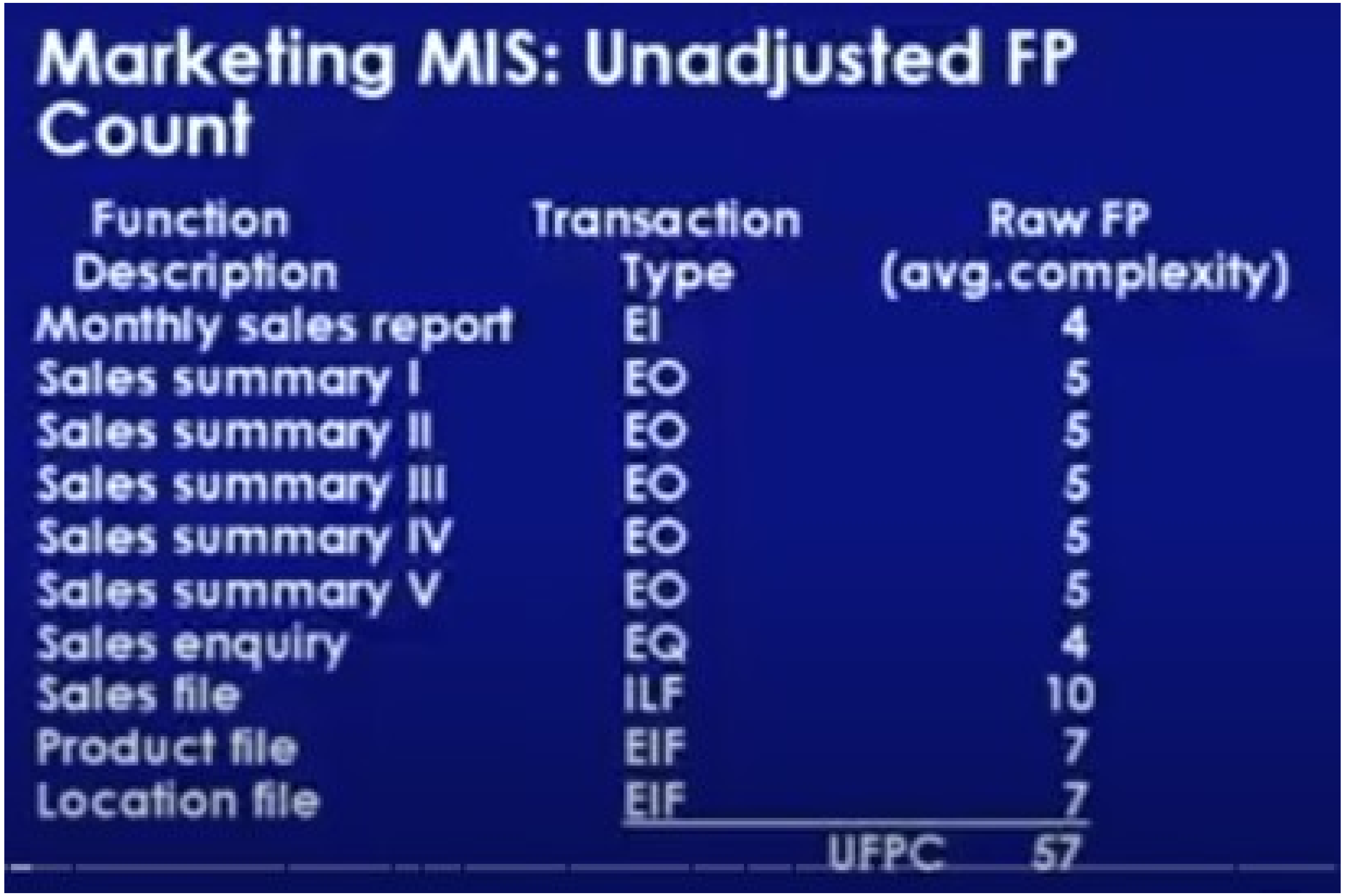


Factor	Value
Backup and recovery	4
Data communications	2
Distributed processing	0
Performance critical	4
Existing operating environment	3
Online data entry	4
Input transaction over multiple screens	5
Master files updated online	3
Information domain values complex	5
Internal processing complex	5
Code designed for reuse	4
Conversion/installation in design	3
Multiple installations	5
Application designed for change	5
Value adjustment factor	**1.17**

Finally, the estimated number of FP is derived:

$$FP_{\text{estimated}} = \text{count total} \times [0.65 + 0.01 \times \Sigma(F_i)] = 375$$

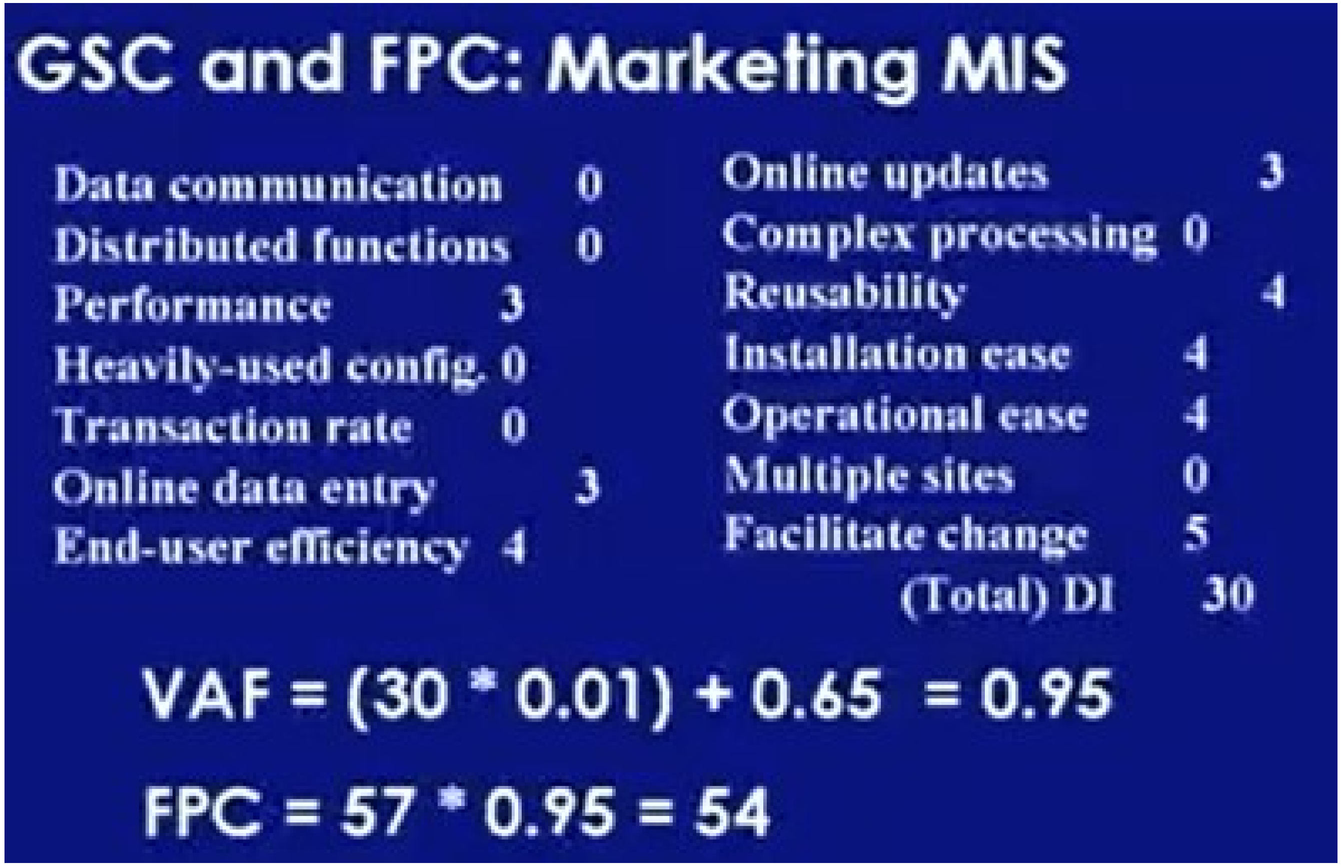
Example : MIS software (FP based)



Marketing MIS: Unadjusted FP Count

Function Description	Transaction Type	Raw FP (avg.complexity)
Monthly sales report	EI	4
Sales summary I	EO	5
Sales summary II	EO	5
Sales summary III	EO	5
Sales summary IV	EO	5
Sales summary V	EO	5
Sales enquiry	EQ	4
Sales file	ILF	10
Product file	EIF	7
Location file	EIF	7
	UFPC	57

14 Adjustment factors of MIS:



GSC and FPC: Marketing MIS

Data communication	0	Online updates	3
Distributed functions	0	Complex processing	0
Performance	3	Reusability	4
Heavily-used config.	0	Installation ease	4
Transaction rate	0	Operational ease	4
Online data entry	3	Multiple sites	0
End-user efficiency	4	Facilitate change	5
		(Total) DI	30

VAF = (30 * 0.01) + 0.65 = 0.95

FPC = 57 * 0.95 = 54

LOC vs FP Based Estimation

LOC and FP estimation techniques differ in the **level of detail required** for decomposition and the target of the partitioning.

LOC estimation is based on Lines of Code which differs based on programmin language

The greater the degree of partitioning, the more likely reasonably accura estimates of LOC can be developed.

For FP estimates, decomposition works differently.

Rather than focusing on functions, each of the **information doma characteristics** is considered.

- **input,outputs, data files, inquiries, and external interfaces.**

14 value adjustment factors

LOC vs FP Based Estimation

FP	LOC
1. FP is specification based.	1. LOC is an analogy(comparison) based.
2. FP is language independent.	2. LOC is language dependent.
3. FP is user-oriented.	3. LOC is design-oriented.
4. It is extendible to LOC.	4. It is convertible to FP (backfiring)

Process Based Estimation

Estimates of effort (in person-months) for each software engineering activity are provided for each CAD software function .

The engineering and construction release activities are subdivided into the major software engineering tasks shown.

Gross estimates of effort are provided for customer communication, planning, and risk analysis. These are noted in the total row at the bottom of the table. Horizontal and vertical totals provide an indication of estimated effort required for analysis, design, code, and test.

The process is divided in to framework activities and each framework activity is divided in to tasks.

Process Based Estimation

A series of framework activities must be performed for each function.

Functions and related framework activities may be represented as pa of a table.

Estimate the effort (e.g., person-months) that will be required accomplish each software process activity for each software function.

Central Matrix

Average labour rates (i.e., cost/unit effort) are then applied to the effo estimated for each process activity.

Labor rate will vary for each task.

Senior staff are heavily involved in early framework activities.

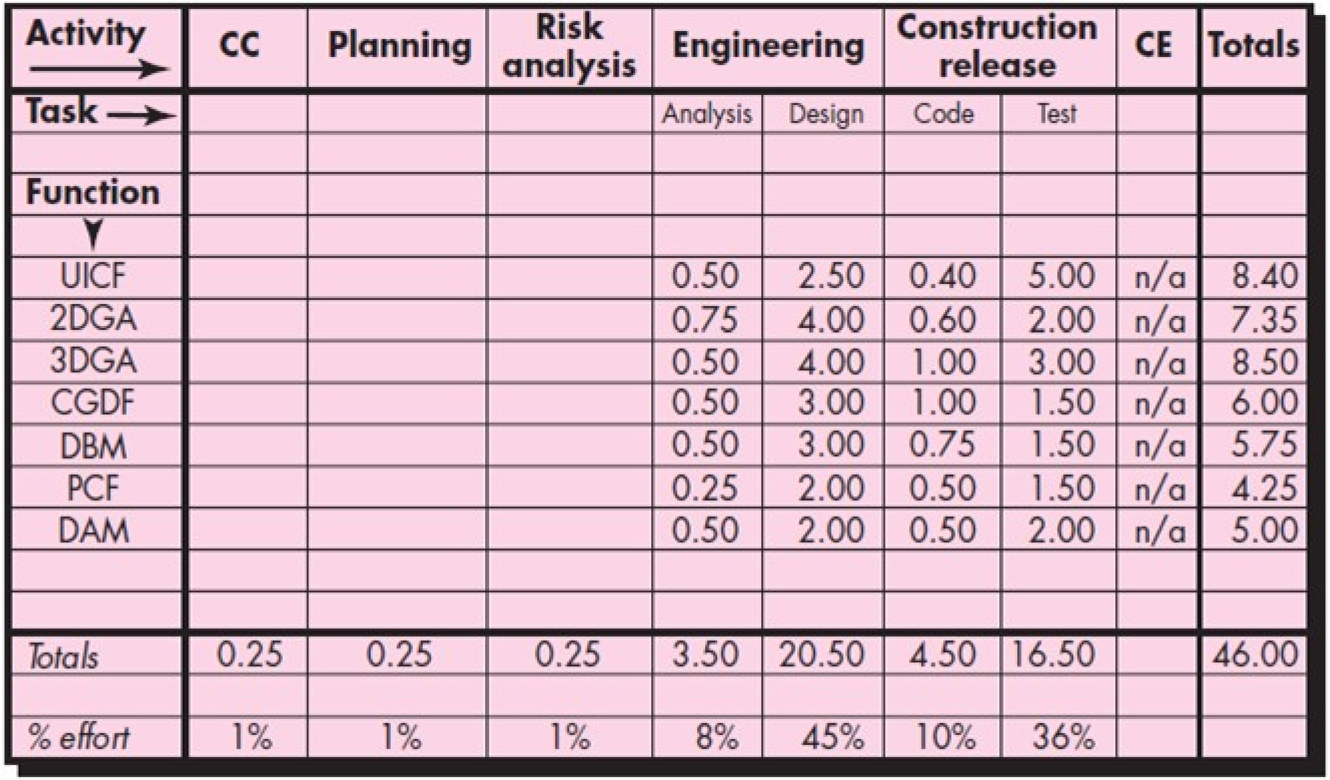
Junior staff will be involved in construction and release.

Process Based Estimation

Most common technique for **estimating a project** is to **base the estimate on the process** that will be used.

The process is **decomposed** into a relatively **small set of tasks** and **effort** required to **accomplish each task** is estimated.

Process-based estimation table



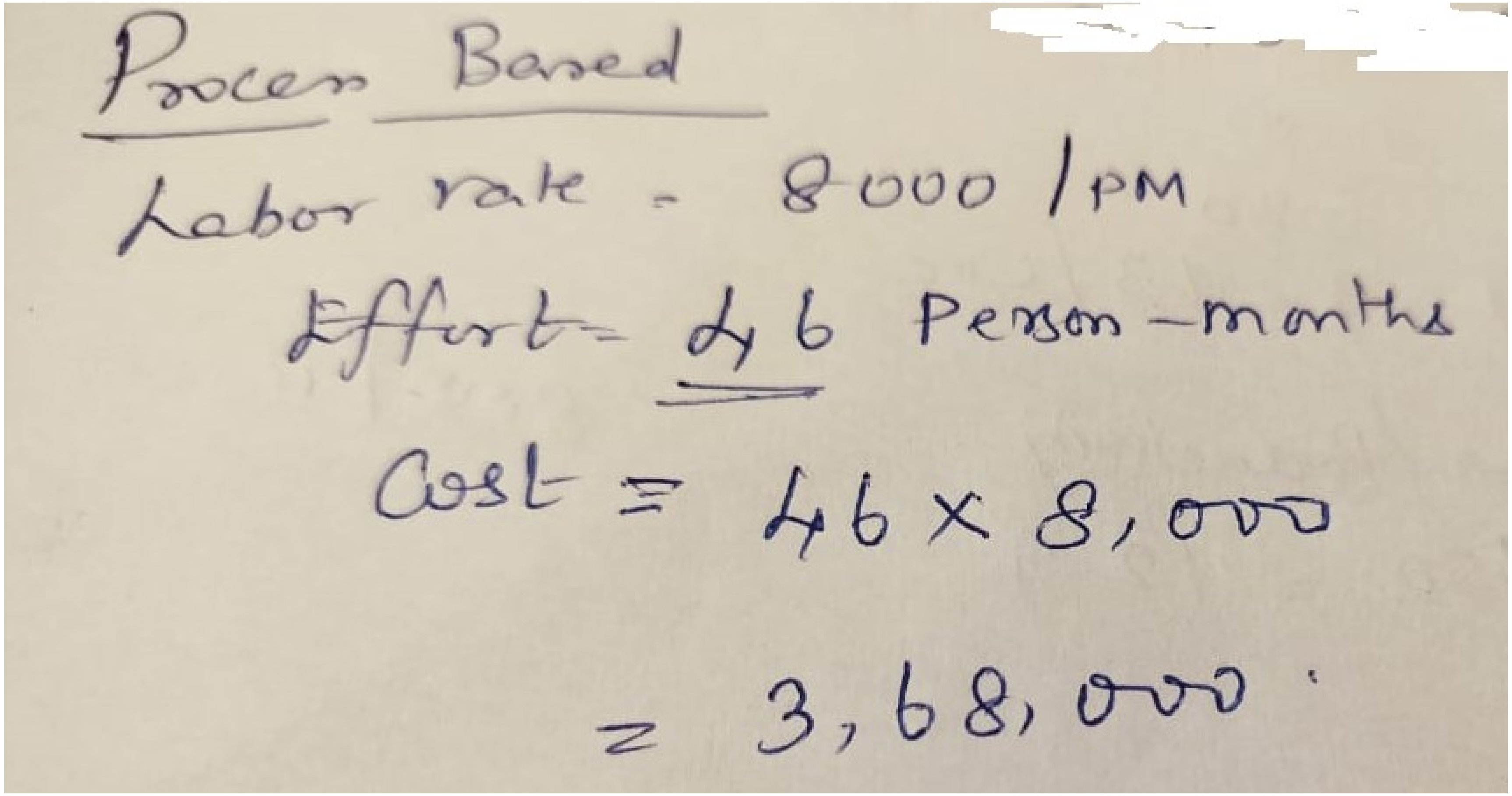
Activity →	CC	Planning	Risk analysis	Engineering		Construction release		CE	Totals
Task →				Analysis	Design	Code	Test		
Function									
↓									
UICF				0.50	2.50	0.40	5.00	n/a	8.40
2DGA				0.75	4.00	0.60	2.00	n/a	7.35
3DGA				0.50	4.00	1.00	3.00	n/a	8.50
CGDF				0.50	3.00	1.00	1.50	n/a	6.00
DBM				0.50	3.00	0.75	1.50	n/a	5.75
PCF				0.25	2.00	0.50	1.50	n/a	4.25
DAM				0.50	2.00	0.50	2.00	n/a	5.00
Totals	0.25	0.25	0.25	3.50	20.50	4.50	16.50		46.00
% effort	1%	1%	1%	8%	45%	10%	36%		

CC = customer communication   CE = customer evaluation

Process Based Estimation

- It should be noted that 53 percent of all effort is expended on front-end engineering tasks (requirements analysis and design), indicating the relative importance of this work.
- Based on an average burdened labor rate of $8,000 per month, the total estimated project cost is $368,000 and the estimated effort is 46 person-months. If desired, labor rates could be associated with each framework activity or software engineering task and computed separately.

olution for Process Based Estimation



Process Based

Labor rate - 8000 / PM

Effort = 46 Person-months

Cost = 46 × 8,000

= 3,68,000

FAQs

It is given that the complexity weighting factors for I, O, E, F, and N are 4, 5, 4, 10, and 7, spectively. It is also given that, out of fourteen value adjustment factors that influence the evelopment effort, four factors are not applicable, each of the other four factors has value 3 nd each of the remaining factors has value 4. The computed value of the function point metr _____. [GATE CS 2015]

(A) 612.06
(B) 404.66
(C) 305.09
(D) 806.9

Solution: Correct Answer is (B).

2. While estimating the cost of the software, Lines of Code(LOC) and Function Points (FP) a used to measure which of the following?

(A) Length of Code

(B) Size of Software

(C) Functionality of Software

(D) None of the Above

Solution: Correct Answer is (B).

FAQs

- **3. In functional point analysis, the number of complexity adjustment factors is**
- **(A)** 10
- **(B)** 12
- **(C)** 14
- **(D)** 20
- **Solution:** Correct Answer is **(C).**
- **FAQs :**
- **1. What do you mean by Functional Point?**
- Ans: Functional Point basically determines the size of the application system on the basis of the functionality of the system.

FAQs

2. How do you find the Functional Point?

- Ans: The functional Point is calculated with the total count factor. It is simply calculated using the formula $FP = TC * [0.65 + 0.01*\sum(Xi)]$.

3. List the five components of the Functional Point?

- Ans: The five components of the functional point are listed below:
- Internal Logical Files (ILF)
- External Interface Files (EIF)
- External Inputs (EI)
- External Outputs (EO)
- External Enquiries (EQ)