Cost Estimation – Predictive Maintenance System for Industrial Machinery

Group Members

- 202101040041 - Tanavi Prakash Gaikwad (A1)
- 202101040039 - Pritam Rajendra Gajare (A1)
- 202101040135 - Nandini Chandrakant Gangurde (B2)
- 202101040137 - Gauri Vinod Kardekar (B2)
- 202101040262 - Manish Jitendra Meshram (C3)
- 202101070082 - Mrinal Gupta (C3)

COCOMO Model:

Intermediate Model

Development mode:

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.

Function Points:

<u>Step 1:</u>

Function point analysis measures the size of the software in terms of the amount of functionality in a system. Function points are computed by first calculating an unadjusted function point count (UFC). Counts are made for the following categories

External inputs: those items provided by the user that describe distinct application- oriented data (such as file names and menu selections)

External outputs: those items provided to the user that generate distinct application-oriented data (such as reports and messages, rather than the individual components of these)

External requests: interactive inputs requiring a response

External interfaces: machine-readable interfaces to other systems

Internal files: logical master files in the system

Unadjusted Function Point Count for Realtime Collaborative Editor

Parameter	Value
User inputs	11
User outputs	1
User requests	0
Files	2
External interfaces	1

Step 2:

Multiply each number by a **weight factor**, according to the complexity (**simple**, **average** or **complex**) of the parameter, associated with that number. The value is given by a table:

Parameter	simple	average	complex
users inputs	3	4	6
users outputs	4	5	7
users requests	3	4	6
files	7	10	15
external interfaces	5	7	10

weight factor

Parameter	Value	Weight Factor
User inputs	11	3 (simple)
User outputs	1	4 (simple)
User requests	0	3 (simple)
Files	2	7 (simple)

External interfaces	1	5 (simple)

Step 3:

Calculate the total UFP (Unadjusted Function Points)

Total UFP = (11x3) + (1x4) + (0x3) + (2x7) + (1x5)

Total UFP = 56

Step 4:

Calculate the DI (degree of influence) by giving a value between 0 and 5 according to the importance of the following points:

Technical Complexity Factors:

Sr. No.	**Technical Complexity Factor**	**Value**
1.	Data Communication	4
2.	Distributed Data Processing	4
3.	Performance Criteria	4
4.	Heavily Utilized Hardware	5
5.	High Transaction Rates	2
6.	Online Data Entry	3
7.	Online Updating	1
8.	End-user Efficiency	3
9.	Complex Computations	4
10.	Reusability	4
11.	Ease of Installation	2
12.	Ease of Operation	3
13.	Portability	3
14.	Maintainability	4

Step 5:

Sum the resulting numbers to obtain **DI** (degree of influence)

Degree of Influence (DI) = 4+4+4+5+2+3+1+3+4+4+2+3+3+4 = **46**

Step 6:

Calculate TCF (Technical Complexity Factor) by given by the formula:

TCF=0.65+0.01*DI

TCF = 0.65 + 0.01 x 46

TCF = 1.11

Step 7:

Function Points are given by the formula:

FP = UFP*TCF

FP = 56 x 1.11 = **62.16**

LOC per FP for Python = 21

$$\textbf{KLOC} = \frac{\text{FP x LOC per FP}}{1000}$$

$$\text{KLOC} = \frac{62.16 \times 21}{1000}$$

KLOC = 1.305 KLOC

Step 8:

Intermediate COCOMO Model formula for effort:

$E = aKLOC^b \times EAF$

For organic mode,

$a = 3.2$, $b = 1.05$

Effort Adjustment Factor (EAF):

Total EAF = Product of the selected factors

Cost Drivers	Ver	Lo	Nomin	Hig	Ver	Extr
Required Reliability	.75	.88	1.00	1.1	1.4	1.40
Database Size	.94	.94	1.00	1.0	1.1	1.16
Product Complexity	.70	.85	1.00	1.1	1.3	1.65
Execution Time	1.0	1.0	1.00	1.1	1.3	1.66
Main Storage	1.0	1.0	1.00	1.0	1.2	1.56
Virtual Machine	.87	.87	1.00	1.1	1.3	1.30
Comp Turn Around	.87	.87	1.00	1.0	1.1	1.15
Analyst Capability	1.4	1.1	1.00	.86	.71	.71
Application Experience	1.2	1.1	1.00	.91	.82	.82
Programmers	1.4	1.1	1.00	.86	.70	.70
Virtual machine	1.2	1.1	1.00	.90	.90	.90
Language Experience	1.1	1.0	1.00	.95	.95	.95
Modern Prog Practices	1.2	1.1	1.00	.91	.82	.82
SW Tools	1.2	1.1	1.00	.91	.83	.83
Required Dev Schedule	1.2	1.0	1.00	1.0	1.1	1,10

EAF = 1.1x1.16x1.00x1.00x1.0x0.87x1.00x0.71x1.00x1.2x1.1x1.2x0.91x1.00 EAF = 1.136 EAF

$PM = aKLOC^b$

PM $= 3.2 \times (1.305)^{1.05} =$ **4.23**

Effort = PM x EAF

Effort = 4.23 x 1.136 = 4.805

Step 9:

Adjusted value of Effort: Adjusted Person Months:

APM = PM * (Total EAF)

APM = 4.23 x 1.136

APM = 4.805

Step 10:

Development Time, **TDEV = C * (APM **D)** months.

For organic mode,

C = 2.5

D = 0.38

TDEV = 2.5 x $(4.805)^{0.38}$

TDEV = 4.53 months

Step 11:

Number of Personnel, NP = APM / TDEV

NP = 4.804/4.53

NP = 1.06 FSP (Full time-equivalent Software Personnel)

Distribution of Effort:

A development process typically consists of the following stages:

- Requirements Analysis
- Design (High Level + Detailed)
- Implementation & Coding
- Testing (Unit + Integration)

Percentage Distribution of Effort and Time Table:

	Req Analysis	Design, HLD + DD	Implementation	Testing	
Effort	**23%**	**29%**	**22%**	**21%**	**100%**
TDEV	**39%**	**25%**	**15%**	**21%**	**100%**

Error Estimation:

Calculate the estimated number of errors i.e., total errors found in requirements, specifications, code, user manuals, and bad fixes.

Adjust the Function Point:

AFP = FP ** 1.25

AFP = $62.16^{1.25}$ = 174.53

Use the following table for calculating error estimates:

Error Type	Error / AFP
Requirements	1
Design	1.25
Implementation	1.75
Documentation	0.6
Due to Bug Fixes	0.4

Errors in Requirements = 174.53 x 1 = 174.53

Errors in Design = 174.53 x 1.25 = 218.16

Errors in Implementation = 174.53 x 1.75 = 305.42

Errors in Documentation = 174.53 x 0.6 = 104.71

Errors in Bug Fixes = 174.53 x 0.4 = 69.81

Productivity:

$$\text{Productivity} = \frac{\text{KLOC}}{\text{Effort}}$$

$$= \frac{1.305}{4.805}$$

$$= 0.271 \text{ KLOC/PM}$$

$$= 271 \text{ LOC/PM}$$