EX:2 Calculate EFFORT using FUNCTION POINT oriented estimation model

AIM:-To Calculate EFFORT using FUNCTION POINT oriented estimation model

Description:-

Function Point (FP) is a metric used in software development to measure the functionality provided by

a software application.

Function Point Analysis (FPA) is a method that uses Function Points to estimate the effort required

for software development.

The effort estimation is typically calculated using the following formula:

Effort=FP×C

where: FP is the total Function Points calculated for the project.

C is the productivity constant, representing the average effort required per Function Point. C

is expressed in person-months per Function Point.

The value of C depends on various factors such as the complexity of the project, the skills and

experience of the development team, the development environment, and so on.

C is determined based on historical data from similar projects or industry

benchmarks.

C= Productivity Factor/100

The total effort estimation in person-months can be obtained by multiplying FP by C

Keep in mind that this is a high-level estimation, and actual project circumstances may vary,

influencing the final effort required.

Procedure :-

To perform a Function Point Analysis and effort estimation, typically follow these steps:

1. Identify and classify the functions in the software.

Table-1:

1.Number of USER INPUTS (EI)

2. Number of Output (EO)

3. Number of ENQUIRIES (EQ)

4. Number of LOGIC FILES (ILF)

5. Number of interfaces (EIF)

2. Assign complexity weights to each function type. WTF

Measurement Parameter Low Average High :

table-2:

1. Number of USER INPUTS (WEI) 3 4 6

2. Number of OUTPUTS (WEO) 4 5 7

3. Number of ENQUIERIES (WEQ) 3 4 6

4. Number of LOGIC FILES (WILF) 7 10 15

5. Number of interfaces (WEIF) 5 7 10

3. Calculate the Unadjusted Function Points (UFP) by summing up the weighted values.

UFP = (EI)\*WEI+ (EO)\*WEO+ (EQ)\*WEQ + (ELF)\*WELF+ (EILF)\*WEILF

4. Calculate Complexity Adjustment Factor (CAF).

CAF =[0.65 +0.01 Σ(Fi)]

The Fi(i=1 to 14) are "complexity adjustment values" based on responses to the following questions:

1. Does the system requine reliable backup and recovery?

2. Are data communications required?

3. Are there distributed processing functions?

4. Is performance critical?

5. Wil the system run in an existing, heavily utilized operational environment?

6 . Does the system require on-line data entry?

7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?

8. Are the master files updated on-line?

9 . Are the inputs, outputs, files, or inquiries comples?

10. Is the internal processing complex?

11. is the code designed to be reusable?

12. Are convension 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?

0 - No Influence

1 - Incidental

2 - Moderate

3 - Average

4 - Significant

5 - Essential

5. FP=UCF X CAF

6. Determine the productivity constant C based on historical data or industry benchmarks.

7. Calculate the effort using the formula Effort=FP×C.

Example: Online Shopping System

Consider an online shopping system with the following identified functions:

Number of user input EI: 5

Number of user output EO: 5

Number of user enquires EQ: 6

Number of Logic files ELF :5

Number of interfaces EILF : 5

Assuming weight factors

USER INPUTS (WEI)=4

OUTPUTS (WEO)=5

ENQUIERIES (WEQ)=4

LOGIC FILES (WILF)=10

interfaces (WEIF)WEI=10

Unadjusted Function Points UFP = (EI)\*WEI+ (EO)\*WEO+ (EQ)\*WEQ + (ELF)\*WELF+

(EILF)\*WEILF

UFP= 5X4+5X5 +6X4 +5X10 +5X10=169

COMPLEXITY ADJUSTMENT FACTOR (CAF)= 0.65 +0.01 X F

Assuming F=25

CAF=0.9

FUNCTION POINT (FP) = UFP X CAF= 152.1

Effort = FP X C =152.1 X C

if C=20

Effort =30.42 Person-month.

C-Program To Calculate EFFORT using FUNCTION POINT oriented estimation

model

#include <stdio.h>

// Function to calculate Function Points with weights

float calfp (int ei, int eo, int ilf, int eif, int eq,

float wei, float weo, float wilf, float weif, float weq, int F)

{

float ufp,caf;

ufp= (ei \* wei) + (eo \* weo) + (ilf \* wilf) + (eif \* weif) + (eq \* weq);

caf=0.65 + (0.01 \*F);

return ufp\*caf;

}

float caleffort(float functionPoints, float productivity)

{

return functionPoints \* productivity/100;

}

int main() {

int ei, eo, ilf, eif, eq,FV;

printf ("\n\*\*\*\*\*\*\*\*ENTER DETAILS OF PROJECT COMPONENTS\*\*\*\*\*\*");

printf("\nEnter the number of USER INPUTS (EI): ");

scanf("%d", &ei);

printf("\nEnter the number of USER OUTPUTS (EO): ");

scanf("%d", &eo);

printf("\nEnter the number of Internal Logic Files (ILF): ");

scanf("%d", &ilf);

printf("\nEnter the number of Interface Files (EIF): ");

scanf("%d", &eif);

printf("\nEnter the number of Inquiries (EQ): ");

scanf("%d", &eq);

float wei, weo, wilf, weif, weq;

printf("\n ENTER COMPONENT WEIGHT FACTORS FOR ABOVE........:");

printf("\nEnter the weight for USER INPUTS (WEI): ");

scanf("%f", &wei);

printf("\nEnter the weight for USER Outputs (WEO): ");

scanf("%f", &weo);

printf("Enter the weight for LOGIC FILES (WILF): ");

scanf("%f", &wilf);

printf("\nEnter the weight for Interface Files (WEIF): ");

scanf("%f", &weif);

printf("\nEnter the weight for Inquiries (WEQ): ");

scanf("%f", &weq);

printf("\n COMPLEXITY ADJUSTMENT VALUE (F)=");

scanf("%d",&FV);

float fp = calfp(ei, eo, ilf, eif, eq, wei, weo, wilf, weif, weq,FV);

float C;

printf("\n Enter the productivity factor: ");

scanf("%f", &C);

float effort = caleffort(fp, C);

printf("\n \*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*");

printf("\n Function Points: %.2f\n", fp);

printf("\n Effort estimation: %.2f person-months\n", effort);

return 0;

}

OUT PUT:-

RESULT :-