DELHI TECHNOLOGICAL UNIVERSITY

Software Project Management

[SE-427]

LAB FILE

Submitted by

Ayush Kumar

(2K19/SE/026)

EXPERIMENT-1

AIM: - Write a program to implement function point analysis and calculate productivity, quality, cost and documentation of the project.

THEORY: -

FPA provides a standardized method to functionally size the software work product. This work product is the output of software new development and improvement projects for subsequent releases. It is the software that is relocated to the production application at project implementation. It measures functionality from the user's point of view i.e., on the basis of what the user requests and receives in return.

Function Point Analysis (FPA) is a method or set of rules of Functional Size Measurement. It assesses the functionality delivered to its users, based on the user's external view of the functional requirements. It measures the logical view of an application, not the physically implemented view or the internal technical view.

The Function Point Analysis technique is used to analyse the functionality delivered by software and Unadjusted Function Point (UFP) is the unit of measurement.

Objectives of FPA:

- The objective of FPA is to measure the functionality that the user requests and receives.
- The objective of FPA is to measure software development and maintenance independently of the technology used for implementation.
- It should be simple enough to minimize the overhead of the measurement process.
- It should be a consistent measure among various projects and organizations.

Types of FPA:

- **Transactional Functional Type –**
    - **External Input (EI):** EI processes data or control information that comes from outside the application's boundary. The EI is an elementary process.
    - **External Output (EO):** EO is an elementary process that generates data or control information sent outside the application's boundary.
    - **External Inquiries (EQ):** EQ is an elementary process made up of an input-output combination that results in data retrieval.
- **Data Functional Type –**
    - **Internal Logical File (ILF):** A user identifiable group of logically related data or control information maintained within the boundary of the application.
    - **External Interface File (EIF):** A group of users recognizable logically related data allusion to the software but maintained within the boundary of another software.

CODE:

```
#include <bits/stdc++.h>
using namespace std;

long long int calculateUfp(vector<vector<int>> &wf, int ui, int uo, int ue, int uf, int ei)
{
    long long int ufp = 0;
    ufp = wf[0][1] * ui + wf[1][1] * uo + wf[2][1] * ue + wf[3][1] * uf + wf[4][1] * ei;
    return ufp;
}

double calculateCAF(double c, double m, int q, vector<int> &rf)
{
    return (c + m * (q * rf[3]));
}


int main()
{
    int questions;

    cout << "Number of questions: ";
    cin >> questions;

    cout << "Weight Factors: " << endl;

    vector<vector<int>> weightingFactors(5, vector<int>(3));

    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            cin >> weightingFactors[i][j];
        }
    }
    cout << endl;

    vector<int> rateFactors(6);

    cout << "Rate Factors: " << endl;
    for (int i = 0; i < 6; i++)
        cin >> rateFactors[i];
    cout << endl;

    int ui, uo, ue, uf, ei;

    cout << "Enter User inputs, user output, user enquiries, user files, external interfaces respectively: ";
    cin >> ui >> uo >> ue >> uf >> ei;

    cout << endl;

    long long int UFP = calculateUfp(weightingFactors, ui, uo, ue, uf, ei);
    cout << "UFP: " << UFP << endl;

    double CAF = calculateCAF(0.65, 0.01, questions, rateFactors);
    double FP = UFP * CAF;

    cout << "CAF: " << CAF << endl;
    cout << "Functional Point: " << FP << endl;
```

```
    int person_permonth;
    int defects;
    int rupees;
    int pages;

    cout << "Person Permonth, Defects, Rupees,pages " << endl;
    cin >> person_permonth >> defects >> rupees >> pages;

    double productivity = FP / person_permonth;
    double Quality = defects / FP;
    double cost = rupees / FP;
    double documentaion = pages / FP;

    cout << "Productivity: " << productivity << endl;
    cout << "Quality: " << productivity << endl;
    cout << "Cost: " << cost << endl;
    cout << "Documentaion: " << documentaion << endl;
    cout<<endl;

    return 0;
}
```

OUTPUT:

```
PS D:\Labs\SPM> cd "d:\Labs\SPM\" ; if ($?) { g++ first.cpp -o first } ; if ($?) { .\first }
Number of questions: 14
Weight Factors:
3 4 6
4 5 7
3 4 6
7 10 15
5 7 10

Rate Factors:
0 1 2 3 4 5

Enter User inputs, user output, user enquiries, user files, external interfaces respectively: 50 40 35 6 4

UFP: 628
CAF: 1.07
Functional Point: 671.96
Person Permonth, Defects, Rupees,pages
100 200 10000 500
Productivity: 6.7196
Quality: 6.7196
Cost: 14.8818
Documentaion: 0.744092
```

CONCLUSION: - Successfully implemented function point analysis and calculated productivity, quality, cost and documentation of the project.

EXPERIMENT-2

AIM: - Implement the Walston-Felix model and SEL model to estimate LOC, development duration and productivity when expected effort is provided. Compare the performance of both the models.

THEORY: -

Below are the two models in estimating the cost of a software project:

In a static model, a single variable is grabbed as a key element for calculating the cost and effort whereas, in a dynamic model, all variables are connected with each other, and there is no primary variable.

1. Static Single Variable model- The Methods using this model utilizes an equation to get the desired values such as cost, time, and effort, etc. And these all depend on the same variable used as a predictor like, size. Below is the example of the most common equation:

$$C = aL^b$$

Where C is cost, L is size and a, b are constants.

We have an example of the static single variable model, i.e. **SEL model** which is used for estimating software production. The equation of this model is given below:

$$E = 1.4L^{0.93}$$
$$DOC = 30.4L^{0.90}$$
$$D = 4.6L^{0.26}$$

Where E is in Person-months, DOC i.e., documentation is in the number of pages, D is duration which is months.

2. Static Multivariable model- These models are also known as **multivariable models.** This model is often based on the first equation and actually depends on several variables representing different aspects of the software development environment. Equations are:

$$E = 5.2L^{0.91}$$

$$D = 4.1L^{0.36}$$

Where E is in Person-months, D is duration which is months.

CODE:

```cpp
#include <bits/stdc++.h>
using namespace std;

double L(double a, double b, double E)
{
    double res = pow(E / a, 1 / b);
    return res;
}

double D(double a, double b, int l)
{
    double res = a * pow(double(l), b);
    return res;
}

double P(double loc, double E)
{
    double res = loc * 1000 / E;
    return res;
}

double M(double E, double d)
{
    double res = E / d;
    return res;
}

int main()
{
    double E;

    cout << "Enter Effort(person-month): ";
    cin >> E;

    double lsel = L(1.4, 0.93, E);
    double dsel = D(4.6, 0.26, lsel);
    double psel = P(lsel, E);
    double msel = M(E, dsel);

    cout << "\nFOR SEL MODEL:-" << endl;
    cout << "LOC: " << int(lsel * 1000) << " LOC" << endl;
    cout << "Duration: " << ceil(dsel) << " months" << endl;
    cout << "Productivity: " << psel << " LOC/person-months" << endl;
    cout << "Average Manning: " << msel << " persons" << endl
         << endl;

    double lwf = L(5.2, 0.91, E);
    double dwf = D(4.1, 0.36, lwf);
    double pwf = P(lwf, E);
    double mwf = M(E, dwf);

    cout << "FOR W-F MODEL:-" << endl;
    cout << "LOC: " << int(lwf * 1000) << " LOC" << endl;
    cout << "Duration: " << ceil(dwf) << " months" << endl;
    cout << "Productivity: " << pwf << " LOC/person-months" << endl;
```

```
    cout << "Average Manning: " << mwf << " persons" << endl
         << endl;

    return 0;
}
```

OUTPUT:

```
PS D:\Labs\SPM> cd "d:\Labs\SPM\" ; if ($?) { g++ second.cpp -o second } ; if ($?) { .\second }
Enter Effort(person-month): 96

FOR SEL MODEL:-
LOC: 94264 LOC
Duration: 15 months
Productivity: 981.92 LOC/person-months
Average Manning: 6.40472 persons

FOR W-F MODEL:-
LOC: 24632 LOC
Duration: 13 months
Productivity: 256.585 LOC/person-months
Average Manning: 7.45781 persons
```

CONCLUSION: - SEL model is better than W-F model since LOC, Productivity of SEL is more than the W-F model and also Average manning is lower in case of SEL.