Functional Point (FP) Analysis

- FPA is used to make estimate of the software project, including its testing in terms of functionality or function size of the software product.
- The functional size of the product is measured in terms of the function point, which is a standard of measurement to measure the software application.
- The basic and primary purpose of the functional point analysis is to measure and provide the software application functional size to the client, customer, and the stakeholder on their request.

Following are the points regarding FPs

1. FPs of an application is found out by counting the number and types of functions used in the applications. Various functions used in an application can be put under five types, as shown in Table:

Measurements Parameters	**Examples**
1.Number of External Inputs(EI)	Input screen and tables
2. Number of External Output (EO)	Output screens and reports
3. Number of external inquiries (EQ)	Prompts and interrupts.
4. Number of internal files (ILF)	Databases and directories
5. Number of external interfaces	Shared databases and shared

(EIF)	routines.

All these parameters are then individually assessed for complexity

2. FP characterizes the complexity of the software system and hence can be used to depict the project time and the manpower requirement.

3. The effort required to develop the project depends on what the software does.

4. FP is programming language independent.

5. The five parameters mentioned above are also known as information domain characteristics.

All the parameters mentioned above are assigned some weights that have been experimentally determined and are shown in Table

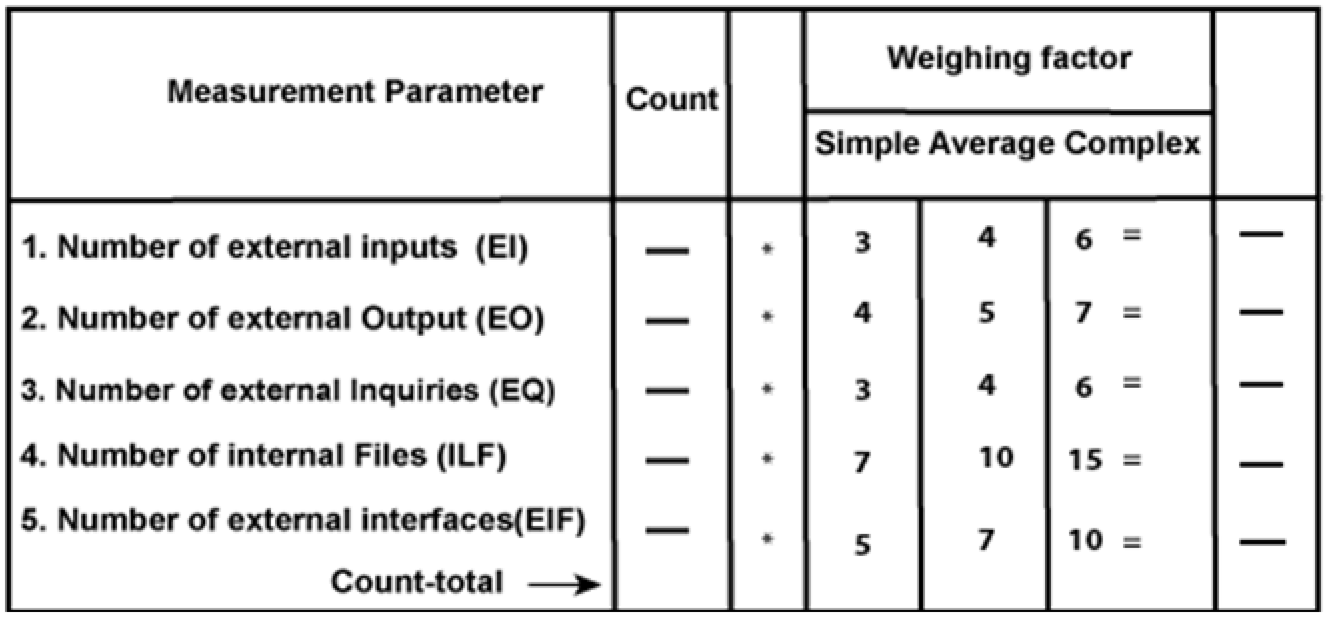
Weights of 5-FP Attributes

Measurement Parameter	Low	Average	High
1. Number of external inputs (EI)	7	10	15
2. Number of external outputs (EO)	5	7	10
3. Number of external inquiries (EQ)	3	4	6

4. Number of internal files (ILF)	4	5	7
5. Number of external interfaces (EIF)	3	4	6

The functional complexities are multiplied with the corresponding weights against each function, and the values are added up to determine the UFP (Unadjusted Function Point) of the subsystem.

Computing FPs



Measurement Parameter	Count		Weighing factor			
			Simple	Average	Complex	
1. Number of external inputs (EI)	—	*	3	4	6 =	—
2. Number of external Output (EO)	—	*	4	5	7 =	—
3. Number of external Inquiries (EQ)	—	*	3	4	6 =	—
4. Number of internal Files (ILF)	—	*	7	10	15 =	—
5. Number of external interfaces(EIF)	—	*	5	7	10 =	—
Count-total →						

Here that weighing factor will be simple, average, or complex for a measurement parameter type.

The Function Point (FP) is thus calculated with the following formula.

FP=Count-total*[0.65+0.01*∑(f_i)]
= Count-total * CAF

where Count-total is obtained from the above Table.

Calculate Complexity Adjustment Factor (CAF)

$$\mathbf{CAF = [0.65 + 0.01 * \sum(f_i)]}$$

and $\mathbf{\sum(f_i)}$ is the sum of all 14 questionnaires and show the complexity adjustment value/ factor-CAF (where i ranges from 1 to 14). Usually, a student is provided with the value of $\sum(f_i)$

Also note that $\mathbf{\sum(f_i)}$ ranges from 0 to 70, i.e.,

$$0 <= \sum(f_i) <=70$$

and CAF ranges from 0.65 to 1.35 because

a. When $\mathbf{\sum(f_i)}$ = 0 then CAF = 0.65
b. When $\mathbf{\sum(f_i)}$ = 70 then CAF = 0.65 + (0.01 * 70) = 0.65 + 0.7 = 1.35

Based on the FP measure of software many other metrics can be computed:

a. Errors/FP
b. $/FP.
c. Defects/FP
d. Pages of documentation/FP
e. Errors/PM.
f. Productivity = FP/PM (effort is measured in person-months).

g. $/Page of Documentation

Example: Compute the function point, productivity, documentation, cost per function for the following data:

1. Number of user inputs = 24
2. Number of user outputs = 46
3. Number of inquiries = 8
4. Number of files = 4
5. Number of external interfaces = 2
6. Effort = 36.9 p-m
7. Technical documents = 265 pages
8. User documents = 122 pages
9. Cost = $7744/ month

Various processing complexity factors are: 4, 1, 0, 3, 3, 5, 4, 4, 3, 3, 2, 2, 4, 5.

Solution:

Measurement Parameter	**Count**		**Weighing factor**
1. Number of external inputs (EI)	24	*	4 = 96
2. Number of external outputs (EO)	46	*	4 = 184
3. Number of external inquiries (EQ)	8	*	6 = 48
4. Number of internal files (ILF)	4	*	10 = 40
5. Number of external interfaces (EIF) Count-	2	*	5 = 10

total →			378

So sum of all f_i ($i \leftarrow 1$ to 14) = 4 + 1 + 0 + 3 + 5 + 4 + 4 + 3 + 3 + 2 + 2 + 4 + 5 = 43

FP=Count-total*[0.65+0.01*∑(f_i)]
=378*[0.65+0.01*43]
=378*[0.65+0.43]
= 378 * 1.08 = 408

$$\text{Productivity} = \frac{\text{FP}}{\text{Effort}} = \frac{408}{36.9} = 11.1$$

Total pages of documentation = technical document + user document
= 265 + 122 = 387pages

Documentation=Pages of documentation/FP
= 387/408 = 0.94

$$\text{Cost per function} = \frac{\text{cost}}{\text{productivity}} = \frac{7744}{11.1} = \$700$$

<u>What is Line of Code (LOC)</u>

1. Line of code (LOC) is a software metric used to measure the size of a computer program by counting the number of lines in the text of the program's source code

2. It is one of the earliest and simpler metrics for calculating the size of the computer program. It is generally used in calculating and comparing the productivity of programmers.