**Functional Point Analysis**

A **Function Point** (FP) is a unit of measurement to express the amount of business functionality, an information system (as a product) provides to a user. FPs measure software size. They are widely accepted as an industry standard for functional sizing.

**IFPUG** − ISO/IEC 20926:2009 Software and systems engineering - Software measurement - IFPUG functional size measurement method.

**Function Point Analysis (FPA) technique** quantifies the functions contained within software in terms that are meaningful to the software users. FPs consider the number of functions being developed based on the requirements specification.

**Function Points (FP) Counting** is governed by a standard set of rules, processes and guidelines as defined by the International Function Point Users Group (IFPUG). These are published in Counting Practices Manual (CPM).

http://www.tutorialspoint.com/estimation\_techniques/index.htm

http://www.tutorialspoint.com/estimation\_techniques/estimation\_techniques\_fp\_counting\_process.htm

https://www.youtube.com/watch?v=vZyUtTH2Bzg

**Elementary Process (EP)**

Elementary Process is the smallest unit of functional user requirement that

* Is meaningful to the user.
* Constitutes a complete transaction.
* Is self-contained and leaves the business of the application being counted in a consistent state.

**Functions**

There are two types of functions −

* Data Functions
* Transaction Functions

Data Functions

There are two types of data functions −

* Internal Logical Files
* External Interface Files

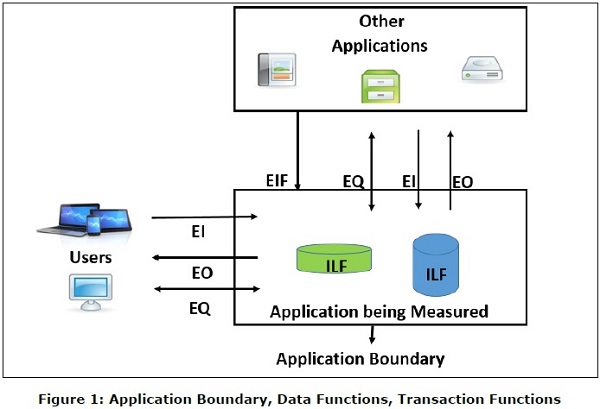
Data Functions are made up of internal and external resources that affect the system.

**Internal Logical Files**

Internal Logical File (ILF) is a user identifiable group of logically related data or control information that resides entirely within the application boundary. The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted. An ILF has the inherent meaning that it is internally maintained, it has some logical structure and it is stored in a file. (Refer Figure 1)

**External Interface Files**

External Interface File (EIF) is a user identifiable group of logically related data or control information that is used by the application for reference purposes only. The data resides entirely outside the application boundary and is maintained in an ILF by another application. An EIF has the inherent meaning that it is externally maintained, an interface has to be developed to get the data from the file. (Refer Figure 1)



**Transaction Functions**

There are three types of transaction functions.

* External Inputs
* External Outputs
* External Inquiries

Transaction functions are made up of the processes that are exchanged between the user, the external applications and the application being measured.

**External Inputs**

External Input (EI) is a transaction function in which Data goes “into” the application from outside the boundary to inside. This data is coming external to the application.

* Data may come from a data input screen or another application.
* An EI is how an application gets information.
* Data can be either control information or business information.
* Data may be used to maintain one or more Internal Logical Files.
* If the data is control information, it does not have to update an Internal Logical File. (Refer Figure 1)

**External Outputs**

External Output (EO) is a transaction function in which data comes “out” of the system. Additionally, an EO may update an ILF. The data creates reports or output files sent to other applications. (Refer Figure 1)

**External Inquiries**

External Inquiry (EQ) is a transaction function with both input and output components that result in data retrieval. (Refer Figure 1)

**Definition of RETs, DETs, FTRs**

**Record Element Type**

A Record Element Type (RET) is the largest user identifiable subgroup of elements within an ILF or an EIF. It is best to look at logical groupings of data to help identify them.

**Data Element Type**

Data Element Type (DET) is the data subgroup within an FTR. They are unique and user identifiable.

**File Type Referenced**

File Type Referenced (FTR) is the largest user identifiable subgroup within the EI, EO, or EQ that is referenced to.

The transaction functions EI, EO, EQ are measured by counting FTRs and DETs that they contain following counting rules. Likewise, data functions ILF and EIF are measured by counting DETs and RETs that they contain following counting rules. The measures of transaction functions and data functions are used in FP counting which results in the functional size or function points.

FP Counting Process involves the following steps −

* **Step 1** − Determine the type of count.
* **Step 2** − Determine the boundary of the count.
* **Step 3** − Identify each Elementary Process (EP) required by the user.
* **Step 4** − Determine the unique EPs.
* **Step 5** − Measure data functions.
* **Step 6** − Measure transactional functions.
* **Step 7** − Calculate functional size (unadjusted function point count).
* **Step 8** − Determine Value Adjustment Factor (VAF).
* **Step 9** − Calculate adjusted function point count.

## Step 1: Determine the Type of Count

There are three types of function point counts −

* Development Function Point Count
* Application Function Point Count
* Enhancement Function Point Count

### Development Function Point Count

Function points can be counted at all phases of a development project from requirement to implementation stage. This type of count is associated with new development work and may include the prototypes, which may have been required as temporary solution, which supports the conversion effort. This type of count is called a baseline function point count.

### Application Function Point Count

Application counts are calculated as the function points delivered, and exclude any conversion effort (prototypes or temporary solutions) and existing functionality that may have existed.

### Enhancement Function Point Count

When changes are made to software after production, they are considered as enhancements. To size such enhancement projects, the Function Point Count gets Added, Changed or Deleted in the Application.

## Step 2: Determine the Boundary of the Count

The boundary indicates the border between the application being measured and the external applications or the user domain. (Refer Figure 1)

To determine the boundary, understand −

* The purpose of the function point count
* Scope of the application being measured
* How and which applications maintain what data
* The business areas that support the applications

## Step 3: Identify Each Elementary Process Required by the User

Compose and/or decompose the functional user requirements into the smallest unit of activity, which satisfies all of the following criteria −

* Is meaningful to the user.
* Constitutes a complete transaction.
* Is self-contained.
* Leaves the business of the application being counted in a consistent state.

For example, the Functional User Requirement − “Maintain Employee information” can be decomposed into smaller activities such as add employee, change employee, delete employee, and inquire about employee.

Each unit of activity thus identified is an Elementary Process (EP).

## Step 4: Determine the Unique Elementary Processes

Comparing two EPs already identified, count them as one EP (same EP) if they −

* Require the same set of DETs.
* Require the same set of FTRs.
* Require the same set of processing logic to complete the EP.

Do not split an EP with multiple forms of processing logic into multiple Eps.

For e.g., if you have identified ‘Add Employee’ as an EP, it should not be divided into two EPs to account for the fact that an employee may or may not have dependents. The EP is still ‘Add Employee’, and there is variation in the processing logic and DETs to account for dependents.

## Step 5: Measure Data Functions

Classify each data function as either an ILF or an EIF.

A data function shall be classified as an −

* Internal Logical File (ILF), if it is maintained by the application being measured.
* External Interface File (EIF) if it is referenced, but not maintained by the application being measured.

ILFs and EIFs can contain business data, control data and rules based data. For example, telephone switching is made of all three types - business data, rule data and control data. Business data is the actual call. Rule data is how the call should be routed through the network, and control data is how the switches communicate with each other.

Consider the following documentation for counting ILFs and EIFs −

* Objectives and constraints for the proposed system.
* Documentation regarding the current system, if such a system exists.
* Documentation of the users’ perceived objectives, problems and needs.
* Data models.

### Step 5.1: Count the DETs for Each Data Function

Apply the following rules to count DETs for ILF/EIF −

* Count a DET for each unique user identifiable, non-repeated field maintained in or retrieved from the ILF or EIF through the execution of an EP.
* Count only those DETs being used by the application that is measured when two or more applications maintain and/or reference the same data function.
* Count a DET for each attribute required by the user to establish a relationship with another ILF or EIF.
* Review related attributes to determine if they are grouped and counted as a single DET or whether they are counted as multiple DETs. Grouping will depend on how the EPs use the attributes within the application.

### Step 5.2: Count the RETs for Each Data Function

Apply the following rules to count RETs for ILF/EIF −

* Count one RET for each data function.
* Count one additional RET for each of the following additional logical sub-groups of DETs.
  + Associative entity with non-key attributes.
  + Sub-type (other than the first sub-type).
  + Attributive entity, in a relationship other than mandatory 1:1.

### Step 5.3: Determine the Functional Complexity for Each Data Function

|  |  |  |  |
| --- | --- | --- | --- |
| **RETS** | **Data Element Types (DETs)** | | |
| **1-19** | **20-50** | **>50** |
| **1** | L | L | A |
| **2 to 5** | L | A | H |
| **>5** | A | H | H |

Functional Complexity: **L** = Low; **A** = Average; **H** = High

### Step 5.4: Measure the Functional Size for Each Data Function

|  |  |  |
| --- | --- | --- |
| **Functional Complexity** | **FP Count for ILF** | **FP Count for EIF** |
| **Low** | 7 | 5 |
| **Average** | 10 | 7 |
| **High** | 15 | 10 |

## Step 6: Measure Transactional Functions

To measure transactional functions following are the necessary steps −

### Step 6.1: Classify each Transactional Function

Transactional functions should be classified as an External Input, External Output or an External Inquiry.

### External Input

External Input (EI) is an Elementary Process that processes data or control information that comes from outside the boundary. The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system.

All of the following rules must be applied −

* The data or control information is received from outside the application boundary.
* At least one ILF is maintained if the data entering the boundary is not control information that alters the behavior of the system.
* For the identified EP, one of the three statements must apply −
  + Processing logic is unique from processing logic performed by other EIs for the application.
  + The set of data elements identified is different from the sets identified for other EIs in the application.
  + ILFs or EIFs referenced are different from the files referenced by the other EIs in the application.

### External Output

External Output (EO) is an Elementary Process that sends data or control information outside the application’s boundary. EO includes additional processing beyond that of an external inquiry.

The primary intent of an EO is to present information to a user through processing logic other than or in addition to the retrieval of data or control information.

The processing logic must −

* Contain at least one mathematical formula or calculation.
* Create derived data.
* Maintain one or more ILFs.
* Alter the behavior of the system.

All of the following rules must be applied −

* Sends data or control information external to the application’s boundary.
* For the identified EP, one of the three statements must apply −
  + Processing logic is unique from the processing logic performed by other EOs for the application.
  + The set of data elements identified are different from other EOs in the application.
  + ILFs or EIFs referenced are different from files referenced by other EOs in the application.

Additionally, one of the following rules must apply −

* The processing logic contains at least one mathematical formula or calculation.
* The processing logic maintains at least one ILF.
* The processing logic alters the behavior of the system.

### External Inquiry

External Inquiry (EQ) is an Elementary Process that sends data or control information outside the boundary. The primary intent of an EQ is to present information to the user through the retrieval of data or control information.

The processing logic contains no mathematical formula or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered.

All of the following rules must be applied −

* Sends data or control information external to the application’s boundary.
* For the identified EP, one of the three statements must apply −
  + Processing logic is unique from the processing logic performed by other EQs for the application.
  + The set of data elements identified are different from other EQs in the application.
  + The ILFs or EIFs referenced are different from the files referenced by other EQs in the application.

Additionally, all of the following rules must apply −

* The processing logic retrieves data or control information from an ILF or EIF.
* The processing logic does not contain mathematical formula or calculation.
* The processing logic does not alter the behavior of the system.
* The processing logic does not maintain an ILF.

### Step 6.2: Count the DETs for Each Transactional Function

Apply the following Rules to count DETs for EIs −

* Review everything that crosses (enters and/or exits) the boundary.
* Count one DET for each unique user identifiable, non-repeated attribute that crosses (enters and/or exits) the boundary during the processing of the transactional function.
* Count only one DET per transactional function for the ability to send an application response message, even if there are multiple messages.
* Count only one DET per transactional function for the ability to initiate action(s) even if there are multiple means to do so.
* Do not count the following items as DETs −
  + Attributes generated within the boundary by a transactional function and saved to an ILF without exiting the boundary.
  + Literals such as report titles, screen or panel identifiers, column headings and attribute titles.
  + Application generated stamps such as date and time attributes.
  + Paging variables, page numbers and positioning information, for e.g., ‘Rows 37 to 54 of 211’.
  + Navigation aids such as the ability to navigate within a list using “previous”, “next”, “first”, “last” and their graphical equivalents.

Apply the following rules to count DETs for EOs/EQs −

* Review everything that crosses (enters and/or exits) the boundary.
* Count one DET for each unique user identifiable, non-repeated attribute that crosses (enters and/or exits) the boundary during the processing of the transactional function.
* Count only one DET per transactional function for the ability to send an application response message, even if there are multiple messages.
* Count only one DET per transactional function for the ability to initiate action(s) even if there are multiple means to do so.
* Do not count the following items as DETs −
  + Attributes generated within the boundary without crossing the boundary.
  + Literals such as report titles, screen or panel identifiers, column headings and attribute titles.
  + Application generated stamps such as date and time attributes.
  + Paging variables, page numbers and positioning information, for e.g., ‘Rows 37 to 54 of 211’.
  + Navigation aids such as the ability to navigate within a list using “previous”, “next”, “first”, “last” and their graphical equivalents.

### Step 6.3: Count the FTRs for Each Transactional Function

Apply the following rules to count FTRs for EIs −

* Count a FTR for each ILF maintained.
* Count a FTR for each ILF or EIF read during the processing of the EI.
* Count only one FTR for each ILF that is both maintained and read.

Apply the following rule to count FTRs for EO/EQs −

* Count a FTR for each ILF or EIF read during the processing of EP.

Additionally, apply the following rules to count FTRs for EOs −

* Count a FTR for each ILF maintained during the processing of EP.
* Count only one FTR for each ILF that is both maintained and read by EP.

### Step 6.4: Determine the Functional Complexity for Each Transactional Function

|  |  |  |  |
| --- | --- | --- | --- |
| **FTRs** | **Data Element Types (DETs)** | | |
| **1-4** | **5-15** | **>=16** |
| **0-1** | L | L | A |
| **2** | L | A | H |
| **>=3** | A | H | H |

Functional Complexity: **L** = Low; **A** = Average; **H** = High

Determine the functional complexity for each EO/EQ, with an exception that EQ must have a minimum of 1 FTR −

|  |  |  |  |
| --- | --- | --- | --- |
| **EQ must have a minimum of 1 FTR**  **FTRs** | **Data Element Types (DETs)** | | |
| 1-4 | 5-15 | >=16 |
| **0-1** | L | L | A |
| **2** | L | A | H |
| **>=3** | A | H | H |

Functional Complexity: **L** = Low; **A** = Average; **H** = High

### Step 6.5: Measure the Functional Size for Each Transactional Function

Measure the functional size for each EI from its functional complexity.

|  |  |
| --- | --- |
| **Complexity** | **FP Count** |
| Low | 3 |
| Average | 4 |
| High | 6 |

Measure the functional size for each EO/EQ from its functional complexity.

|  |  |  |
| --- | --- | --- |
| **Complexity** | **FP Count for EO** | **FP Count for EQ** |
| **Low** | 4 | 3 |
| **Average** | 5 | 4 |
| **High** | 6 | 6 |

## Step 7: Calculate Functional Size (Unadjusted Function Point Count)

To calculate the functional size, one should follow the steps given below −

### Step 7.1

Recollect what you have found in Step 1. Determine the type of count.

### Step 7.2

Calculate the functional size or function point count based on the type.

* For development function point count, go to Step 7.3.
* For application function point count, go to Step 7.4.
* For enhancement function point count, go to Step 7.5.

### Step 7.3

Development Function Point Count consists of two components of functionality −

* Application functionality included in the user requirements for the project.
* Conversion functionality included in the user requirements for the project. Conversion functionality consists of functions provided only at installation to convert data and/or provide other user-specified conversion requirements, such as special conversion reports. For e.g. an existing application may be replaced with a new system.

**DFP = ADD + CFP**

Where,

**DFP** = Development Function Point Count

**ADD** = Size of functions delivered to the user by the development project

**CFP** = Size of the conversion functionality

**ADD** = FP Count (ILFs) + FP Count (EIFs) + FP Count (EIs) + FP Count (EOs) + FP Count (EQs)

**CFP** = FP Count (ILFs) + FP Count (EIFs) + FP Count (EIs) + FP Count (EOs) + FP Count (EQs)

### Step 7.4

Calculate the Application Function Point Count

**AFP = ADD**

Where,

**AFP** = Application Function Point Count

**ADD** = Size of functions delivered to the user by the development project (excluding the size of any conversion functionality), or the functionality that exists whenever the application is counted.

**ADD** = FP Count (ILFs) + FP Count (EIFs) + FP Count (EIs) + FP Count (EOs) + FP Count (EQs)

### Step 7.5

Enhancement Function Point Count considers the following four components of functionality −

* Functionality that is added to the application.
* Functionality that is modified in the Application.
* Conversion functionality.
* Functionality that is deleted from the application.

**EFP = ADD + CHGA + CFP + DEL**

Where,

**EFP** = Enhancement Function Point Count

**ADD** = Size of functions being added by the enhancement project

**CHGA** = Size of functions being changed by the enhancement project

**CFP** = Size of the conversion functionality

**DEL** = Size of functions being deleted by the enhancement project

**ADD** = FP Count (ILFs) + FP Count (EIFs) + FP Count (EIs) + FP Count (EOs) + FP Count (EQs)

**CHGA** = FP Count (ILFs) + FP Count (EIFs) + FP Count (EIs) + FP Count (EOs) + FP Count (EQs)

**CFP** = FP Count (ILFs) + FP Count (EIFs) + FP Count (EIs) + FP Count (EOs) + FP Count (EQs)

**DEL** = FP Count (ILFs) + FP Count (EIFs) + FP COUNT (EIs) + FP Count (EOs) + FP Count (EQs)

### Step 8: Determine the Value Adjustment Factor

GSCs are made optional in CPM 4.3.1 and moved to Appendix. Hence, Step 8 and Step 9 can be skipped.

The Value Adjustment Factor (VAF) is based on 14 GSCs that rate the general functionality of the application being counted. GSCs are user business constraints independent of technology. Each characteristic has associated descriptions to determine the degree of influence.

|  |  |
| --- | --- |
| **General System Characteristic** | **Brief Description** |
| Data Communications | How many communication facilities are there to aid in the transfer or exchange of information with the application or system? |
| Distributed Data Processing | How are distributed data and processing functions handled? |
| Performance | Did the user require response time or throughput? |
| Heavily Used Configuration | How heavily used is the current hardware platform where the application will be executed? |
| Transaction Rate | How frequently are transactions executed daily, weekly, monthly, etc.? |
| On-Line Data Entry | What percentage of the information is entered online? |
| End-user Efficiency | Was the application designed for end-user efficiency? |
| Online Update | How many ILFs are updated by online transaction? |
| Complex Processing | Does the application have extensive logical or mathematical processing? |
| Reusability | Was the application developed to meet one or many user’s needs? |
| Installation Ease | How difficult is conversion and installation? |
| Operational Ease | How effective and/or automated are start-up, back-up, and recovery procedures? |
| Multiple Sites | Was the application specifically designed, developed, and supported to be installed at multiple sites for multiple organizations? |
| Facilitate Change | Was the application specifically designed, developed, and supported to facilitate change? |

The degree of influence range is on a scale of zero to five, from no influence to strong influence.

|  |  |
| --- | --- |
| **Rating** | **Degree of Influence** |
| 0 | Not present, or no influence |
| 1 | Incidental influence |
| 2 | Moderate influence |
| 3 | Average influence |
| 4 | Significant influence |
| 5 | Strong influence throughout |

Determine the degree of influence for each of the 14 GSCs.

The sum of the values of the 14 GSCs thus obtained is termed as Total Degree of Influence (TDI).

**TDI = ∑14 Degrees of Influence**

Next, calculate Value Adjustment Factor (VAF) as

**VAF = (TDI × 0.01) + 0.65**

Each GSC can vary from 0 to 5, TDI can vary from (0 × 14) to (5 × 14), i.e. 0 (when all GSCs are low) to 70 (when all GSCs are high) i.e. 0 ≤ TDI ≤ 70. Hence, VAF can vary in the range from 0.65 (when all GSCs are low) to 1.35 (when all GSCs are high), i.e., 0.65 ≤ VAF ≤ 1.35.

### Step 9: Calculate Adjusted Function Point Count

As per the FPA approach that uses the VAF (CPM versions before V4.3.1), this is determined by,

**Adjusted FP Count = Unadjusted FP Count × VAF**

Where, unadjusted FP count is the functional size that you have calculated in Step 7.

As the VAF can vary from 0.65 to 1.35, the VAF exerts an influence of ±35% on the final adjusted FP count.

## Benefits of Function Points

Function points are useful −

* In measuring the size of the solution instead of the size of the problem.
* As requirements are the only thing needed for function points count.
* As it is independent of technology.
* As it is independent of programming languages.
* In estimating testing projects.
* In estimating overall project costs, schedule and effort.
* In contract negotiations as it provides a method of easier communication with business groups.
* As it quantifies and assigns a value to the actual uses, interfaces, and purposes of the functions in the software.
* In creating ratios with other metrics such as hours, cost, headcount, duration, and other application metrics.

## FP Repositories

International Software Benchmarking Standards Group (ISBSG) grows and maintains two repositories for IT data.

* Development and Enhancement Projects
* Maintenance and Support Applications

There are more than 6,000 projects in the Development and Enhancement Projects repository.

Data is delivered in Microsoft Excel format, making it easier for further analysis that you wish to do with it, or you can even use the data for some other purpose.

ISBSG repository license can be purchased from − <http://www.isbsg.com/>

ISBSG offers 10% discount for IFPUG members for online purchases when the discount code “IFPUGMembers” is used.

ISBSG Software Project Data Release updates can be found at −<http://www.ifpug.org/isbsg/>

COSMIC and IFPUG collaborated to produce a Glossary of terms for software Non-functional and Project Requirements. It can be downloaded from − [cosmic-sizing.org](http://cosmic-sizing.org/publications/glossary-of-terms-for-nf-and-project-requirements/)