An Overview of Function Point Analysis

Function Point Analysis (FPA) is an ISO recognized method to measure the functional size of an information system and a structured method of classifying components of a system. It is a method used to break systems down into smaller components so that they can be better understood and analyzed. Function Point Analysis provides a structured technique for problem solving. The functional size reflects the amount of functionality that is relevant to and recognized by the user in the business. It is independent of the technology used to implement the system.

The unit of measurement is "function points". So, FPA expresses the functional size of an information system in a number of function points (for example: the size of a system is 314 FPs).

The functional size may be used:

- to budget application development or enhancement costs
- to budget the annual maintenance costs of the application portfolio
- to determine project productivity after completion of the project
- to determine the Software Size for cost estimating

In Function Point Analysis, systems are divided into five large classes. The first three classes are External Inputs, External Outputs and External/Data Inquiries. These classes or components transact against files and hence they are also called Transactions. The next two classes are Internal Logical Files and External Interface Files. The data is stored in these classes and they form the Logical Information.

Objectives of Function Point Analysis

Function Points measure systems from a functional perspective and are independent of technology. Hence, regardless of technology, language, development method, or hardware/software used, the number of function points for a system will remain constant. The variable in Function Point Analysis is the amount of effort required to deliver a given set of function points. Therefore, function point analysis can be used to determine which environment / language / tool is more productive.

1. Function point analysis can be used to determine which environment/language/tool is more productive.
2. Function point analysis can provide a mechanism to track and monitor scope creep.
3. Function point counts at the end of each phase/stage of a project can be compared to the function points actually delivered.
4. If the number of function points has increased then there has been a scope creep.
5. The number of function points that has increased or decreased defines the project growth.
6. If the project has grown then it is a sure indication that the requirements haven't been gathered well and there has been some scope creep.
7. If the project growth has declined then the communication between the end-user and the development team has improved.

Process of Counting Function Points

Function point analysis follows specific steps for counting:

- Identification of the subsystem boundaries
- Identification of the data functions (internal logical files and external interface files)
- Identification of transactional functions (external inputs, external outputs and external inquiries)
- Calculation of the Unadjusted Function Point (UAF) Count
- Determination of the Value Adjustment Factor (VAF) using General System Characteristics (GSC)
- Calculation of the final Function Point Count

Establish the Subsystem Boundary

A project may include one or more applications or subsystems. Boundaries must be drawn to identify the external applications that interact with this project or the application that is being measured.

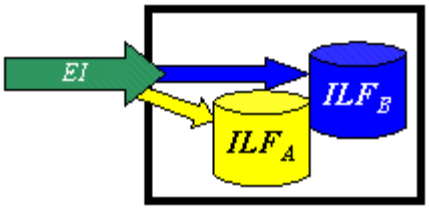
The Five Major Components

Function Point Analysis breaks the systems down into smaller components so that users, developers, and managers can easily understand the functionality of a system; the technical knowledge is not required for the same. In the world of function points, systems are divided into components.

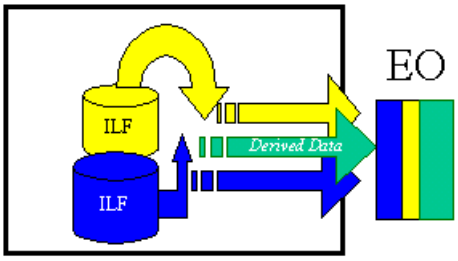
Transactional Functions

The first three components are the systems External Inputs (EI), External Outputs (EO), and External Inquiries (EQ). Each of these components adds, modifies, deletes, retrieves or processes information contained in the files and hence called transactions.

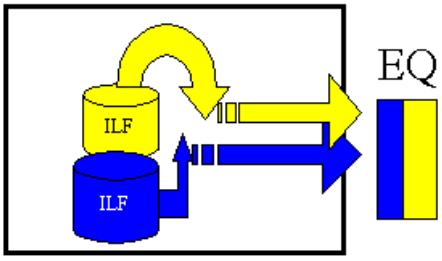
External Inputs – This defines those data that interact with the application from the outside. This data from the outside may be used to maintain internal logical files.



External Outputs – Here, the data passes from inside the application to the outside. The derived data may be used to update internal logical files. The data is used to create reports or output files which, in turn, may be used by other applications. These reports are again created from internal logical files or external interface files.



External Inquiry – This contains both input and output components that retrieve data from internal logical files and external interface files. However, the input here does not update the internal logical files or external interface files as in external inputs, and the output does not contain derived data as in external outputs.



Data Functions

The other two components are the system's files viz., Internal Logical Files (ILF) and External Interface Files (EIF).

Internal Logical Files – This contains logically related data that resides entirely within the application's boundary and is maintained through external inputs.

External Interface Files – This contains logically related data that is used for reference purposes only. The data resides entirely outside the application and is maintained by another application. This means that an external interface file can be an internal logical file of another application.

Rating and Ranking the Components

Once all the components in the application have been classified as one of the five major components mentioned above, they have to be rated as either Low, Average, or High.

Ranking is commonly based on File Types Referenced, Data Element Types and Record Element Types.

File Types Referenced (FTRs) represents the total number of internal logical files (ILFs) maintained, read, or referenced, and the external interface files read or referenced by the EI/EO transaction.

Data Element Type (DET) can be defined as unique user-recognizable non-recursive fields including foreign key attributes that are maintained on ILF/EIF.

Record Element Type (RET) is a subgroup of data elements within an ILF/EIF.

For each of the components belonging to Transactional functions, the ranking is based on:

1. Number of files updated or referenced (FTRs) and
2. Number of data element types (DETs)

For the data components viz., Internal Logical Files (ILF) and External Interface Files (EIF), ranking is based on:

1. Number of Data Element Types (DETs) and
2. Number of Record Element Types (RETs)

The tables given below provide an example of rankings and ratings given to each component.

External Inputs

FTRs	**Data Elements**		
	1-4	**5-15**	**>15**
0-1	Low	Low	Average
2	Low	Average	High
>=3	Average	High	High

External Output/External Inquiries

FTRs	**Data Elements**		
	1-5	**6-19**	**>19**
0-1	Low	Low	Average
2-3	Low	Average	High
>3	Average	High	High

Values or Ratings for Transactions

Ranking	Data Elements		
	External Input	External Output	External Inquiry
Low	3	4	3
Average	4	5	4
High	6	7	6

In the above example, if an external input component updates 2 FTRs and references around 16 data elements then the ranking would be HIGH and the associated rating would be 6.

Rankings for ILFs and EIFs

Record Elements	Data Elements		
	1-19	20-50	>50
1	Low	Low	Average
2-5	Low	Average	High
>5	Average	High	High

Values or Ratings

Ranking	Data Elements	
	Internal Logical File	External Interface File
Low	7	5
Average	10	7
High	15	10

Unadjusted Function Points

Each function point count is multiplied by the numerical rating derived from the above table(s) to determine the complexity rated value. The complexity rated values on each row are summed across the table. This gives the total value for each type of component. The totals for all the components are then summed up to arrive at the total number of Unadjusted Function Points or UAF.

Component Type	Component Complexity			
	Low	Average	High	Total
External Inputs	__* 3=EI1	__* 4=EI2	__* 6=EI3	EI1+EI2+EI3
External Outputs	__* 4=EO1	__* 5=EO2	__* 7=EO3	EO1+EO2+EO3
External Inquiries	__* 3=EQ1	__* 4=EQ2	__* 6=EQ3	EQ1+EQ2+EQ3
Internal Logical Files	__* 7=ILF1	__* 10=ILF2	__* 15=ILF3	ILF1+ILF2+ILF3
External Interface Files	__* 5=EIF1	__* 7=EIF2	__* 10=EIF3	EIF1+EIF2+EIF3
Total UAF	(EI1+EI2+EI3) + (EO1+EO2+EO3) + (EQ1+EQ2+EQ3) + (ILF1+ILF2+ILF3) + (EIF1+EIF2+EIF3)			

Value Adjustment Factor

The Value Adjustment Factor (VAF) is based on fourteen General System Characteristics (GSC's). The degrees of influence range on a scale of zero to five, from no influence to strong influence. The GSCs are listed below:

General System Characteristic		Brief Description
1.	Data communications	How many communication facilities are there to aid in the transfer or exchange of information with the application or system?
2.	Distributed data processing	How are distributed data and processing functions handled?
3.	Performance	Was response time or throughput required by the user?
4.	Heavily used configuration	How heavily used is the current hardware platform where the application will be executed?
5.	Transaction rate	How frequently are transactions executed daily, weekly, monthly, etc.?
6.	On-Line data entry	What percentage of the information is entered On-Line?
7.	End-user efficiency	Was the application designed for end-user efficiency?
8.	On-Line update	How many ILF's are updated by On-Line transaction?
9.	Complex processing	Does the application have extensive logical or mathematical processing?
10.	Reusability	Was the application developed to meet one or many user's needs?
11.	Installation ease	How difficult is conversion and installation?
12.	Operational ease	How effective and/or automated are start-up, back-up, and recovery procedures?
13.	Multiple sites	Was the application specifically designed, developed, and supported to be installed at multiple sites for multiple organizations?
14.	Facilitate change	Was the application specifically designed, developed, and supported to facilitate change?

VAF = 0.65 + (Sum of degrees of Influence of the fourteen GSCs) / 100

The final Function Point Count is obtained by multiplying VAF by the Unadjusted Function Point (UAF).

FP = UAF * VAF

Summary

Below is a short summary of benefits of Function Point Analysis.

- Function Points can be used to size software applications accurately. Sizing is an important component in determining productivity (outputs/inputs).
- Function Points can be an essential ingredient to measuring and managing scope creep.
- Function Points can be the basis of creating estimating models, which can be explained, revised and accurate.
- Function Points can be used with other metrics can help pinpoint opportunities for improvement.
- Function Points can help improve communications with senior management.
- They can be counted by different people, at different times, to obtain the same measure within a reasonable margin of error.
- Function Points are easily understood by the non technical user. This helps communicate sizing information to a user or customer.
- Function Points can be used to determine whether a tool, a language, an environment, is more productive when compared with others.

Accurately predicting the size of software has plagued the software industry for over 45 years. Function Points are becoming widely accepted as the standard metric for measuring software size. Now that Function Points have made adequate sizing possible, it can now be anticipated that the overall rate of progress in software productivity and software quality will improve. Understanding software size is the key to understanding both productivity and quality. Without a reliable sizing metric relative changes in productivity (Function Points per Work Month) or relative changes in quality (Defects per Function Point) can not be calculated. If relative changes in productivity and quality can be calculated and plotted over time, then focus can be put upon an organizations strengths and weaknesses. Most important, any attempt to correct weaknesses can be measured for effectiveness.

Glossary of Terms

DET - Data Element Type
EIF - External Interface File
EI - External Inputs
EO - External Outputs
EQ - External Inquiries
FTR - File Types Referenced
GSC - General System Characteristics
ILF - Internal Logical Files
UAF - Unadjusted Function Point
VAF - Value Adjustment Factor