ISSN: 0975-766X
CODEN: IJPTFI
Available Online through
Research Article
www.ijptonline.com

SOFTWARE DEVELOPMENT EFFORT AND DURATION ESTIMATES USING SIZE BASED FP METHODOLOGY

M.Bala Subramanian*, G. Rajarajeswari**
*Research Scholar, Department of Computer Science, Bharath Institute of Higher Education and Research, Bharath University, Chennai.
**Department of Information technology, RRASE College on Engineering, Chennai.
Email: balamadevan@gmail.com

Received on: 15.10.2016 *Accepted on: 22.11.2016*

Abstract

Today Software is the most expensive component of all computer based system. In particular, predicting the cost of the software has become the most important and critical task. The size of the task of designing and developing a business computerized information system is determined by the product of three factors. The technical complexity factor, processing information size, and the environmental factors. The first two of these factors are in-built to the size of the system in the sense that they result directly from the requirements for the system to be delivered to the user and the for estimation purpose the third group of environmental factors are taken into account. Allan Albrecht has described a method known as "Function Point Analysis" [FPA], for determining the relative size of a system based on the size and TCF that said above. FP is a measure of software size that logically measures the functional terms and the measured size remains same unmodified all through the Software life cycle process. The relative measure of the function values are delivered to the user based on function and based on the end users of the system. In addition to measuring the output, FP analysis is extremely useful in estimating projects, managing change of scope, measuring productivity, and communicating functional requirements. In this paper, it stretches out the basis of the estimation process and a brief literature study on the FP methodology, thru a software tool developed using excel and visual basic to facilitate how the estimation process is carried out in a real time world.

Keywords: Estimation; Software Measurement Size; Function Point Analysis; FPA Calculation.

Introduction Estimation Process

Estimation as process is based on certain set of rules guidelines, which need to be applied consistently to ensure the projects improve the accuracy of estimates over time by calibrating the estimation data based on projects

performance. The estimation methodology may either use "Size" or "Complexity" of work as the basis for estimation. The factors influencing these methodologies are at best defined as guidelines since these may vary from projects to projects based on the project characteristics such as the experience level of the team, technology, application domain and its criticality et al. In the absence of a set of common guidelines, the projects may significantly deviate and the opportunity for identifying best practices to make them available for others in the organization would be totally lost.

In Size (i.e., Function Point) based estimation, the process begins with decomposing the business or system requirements into atomic functions as perceived by the end users.

Function Point Analysis Process, an Introduction

as said in the introduction part, FPA measures software by quantifying the functionality the software provides to the user based primarily on logical design which,

- Measure functionality that the user requests and receives
- Measure software development and maintenance independently of technology used for implementation

FPA Process [9]

The FPA involves with the following process:

- This process gathers all the available documentation and determines the type of count
- Determine counting scope and Boundaries, Identify functional user requirements
- Measure data function/ Transaction function
- Calculate function size
- Determine the VAF value. (Value Adjustment Factor)
- Documents and report.

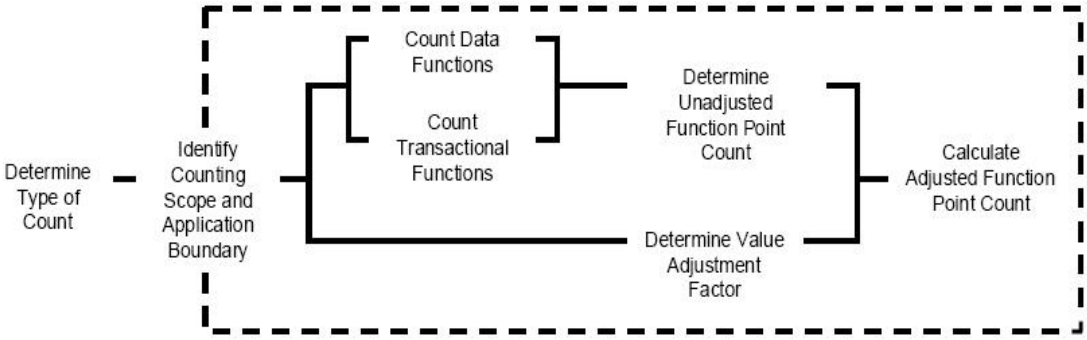
Function Point Counting Documentation [9]

This process gathers all the available documentation and determines the type of count which is based on how it associates with development or Enhancement project function point count.

Types of Function Point Counts:

The first step in the function point counting procedure is to determine the type of function point count based on how it associates with development or Enhancement project function point count.

Function Point Counting Procedures



Counting Scope and Application Boundary Procedures.

When you perform a function point count, the following characteristics of the count should be properly documented:

Step Action

- Establish the purpose of the count
- Identify the counting scope
- Identify the application boundary
- Document the following items:
  a) The purpose of the count
  b) The counting scope
  c) The application boundary
  d) Any assumptions related to the above

The components of Function point Count:

The components are splitted based on two functions:

- Data Functions
  a) Internal Logic Files
  b) External Interface Files
- Transactional functions
  a) External Inputs
  b) External outputs
  c) External inquiries

Data Functions*:* Data functions represent the functionality provided to the user to meet internal and external data requirements. Data function types are defined as internal logical files (ILFs) and external interface files (EIFs).

- *Internal Logical Files (ILF):* It is a user identifiable group of logically related data or control information maintained within the boundary of the application.

- *External Interface Files (EIF):* is a user identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application.

Complexity Measures*:*

The complexity of the ILF\ELF is depends upon:

- Number of data element type (DET)
- Number of record element type (RET)

Procedures:

- Step 1: Use the complexity and contribution counting rules that define above, to identify and count the number of RETs and DETs
- Step 2: Rate the functional complexity.
- Step 3: Translate the ILFs and EIFs to unadjusted function points.
- Step 4: Calculate each ILF and EIF contribution to the unadjusted function point count

Transactional Functions*:*

Transactional functions represent the functionality provided to the user for the processing of data by an application.

Transactional functions are defined as external inputs (EIs), external outputs (EOs), and external inquiries (EQs).

1) External Inputs: An external input (EI) is an elementary process that processes data or control information that comes from outside the application boundary.
2) External Outputs: An external output (EO) is an elementary process that sends data or control information outside the application boundary.
3) External Inquiry: An external inquiry (EQ) is an elementary process that sends data or control information outside the application boundary.

EI/EO/EQ Counting Rules

- Step 5: Identify the elementary processes.
- Step 6: Determine the primary intent of the identified elementary processes, and classify as an EI, EO, or EQ.
- Step 7: Validate against the transaction (EI, EO, EQ) identification rules.
- Step 8: Determine the transaction (EI, EO, EQ) complexity.
- Step 9: Determine the transaction (EI, EO, EQ) contribution to the unadjusted function point count.
- Step 10: Determine the Value Adjustment Factor

	Weighting Factors						
	Functional Complexity Rating with UFP						
Function Point Item Types	Count		Low	Average	High		Total FP
No. of External Inputs (EI)		X	3	4	6	=	
No. of External Output (EO)		X	4	5	7	=	
No. of External Queries		X	3	4	6	=	
No. of Internal Logic Files (ILF)		X	7	10	15	=	
No. of External interfaces files (EIF)		X	5	7	10	=	
Count total = Unadjusted function point count							
Influence Factor (and hence Technical Complexity Factor)							
Adjusted function point count							

The Value Adjustment Factor (*VAF*) is based on 14 general system characteristics (GSCs) that rate the general functionality of the application being counted. Each characteristic has associated descriptions that help determine the degree of influence of that characteristic. (i.e., the degree of influence for each characteristic ranges on a scale of zero to five, from no influence to strong influence). The 14 general system characteristics [8] are summarized into the value adjustment factor. When applied, the value adjustment factor adjusts the unadjusted function point count +/-35 percent to produce the adjusted function point count. This step may be omitted, and unadjusted function points may be used to measure the size of a software application or project.

Procedures:

- Step 1: Evaluate each of the 14 general system characteristics on a scale from zero to five to determine the degree of influence (DI). i.e., a set of 14 questions that evaluate the overall complexity of the application.

The 14 general system characteristics are:

1. Data Communications
2. Distributed Data Processing
3. Performance
4. Heavily Used Configuration
5. Transaction Rate
6. Online Data Entry
7. End-User Efficiency
8. Online Update
9. Complex Processing
10. Reusability
11. Installation Ease
12. Operational Ease
13. Multiple Sites
14. Facilitate Change

Step 2: Add the degrees of influence for all 14 general system characteristics to produce the total degree of influence (TDI). Based on the stated user requirements, each general system characteristic (GSC) must be evaluated in terms of its degree of influence (DI) on a scale of zero to five and the details are as follows:

0	Not present, or no influence
1	Incidental influence
2	Moderate influence
3	Average influence
4	Significant influence
5	Strong influence throughout

Factors determining the Degree of Influence are as follows:

Data Communications

Data Communications describes the degree to which the application communicates directly with the processor.

Distributed Data Processing

Distributed Data Processing describes the degree to which the application transfers data among components of the application.

Performance: Performance describes the degree to which response time and throughput performance considerations influenced the application development.

Heavily Used Configuration

Heavily Used Configuration describes the degree to which computer resource restrictions influenced the development of the application.

Transaction Rate

Transaction Rate describes the degree to which the rate of business transactions influenced the development of the application.

Online Data Entry

Online Data Entry describes the degree to which data is entered through interactive transactions. Online data entry and control functions are provided in the application.

End-User Efficiency

End-User Efficiency describes the degree of consideration for human factors and ease of use for the user of the application measured.

Online Update

Online Update describes the degree to which internal logical files are updated online.

Complex Processing

Complex processing describes the degree to which processing logic influenced the development of the application.

Reusability

Reusability describes the degree to which the application and the code in the application have been specifically designed, developed, and supported to be usable in *other* applications.

Installation Ease

Installation Ease describes the degree to which conversion from previous environments influenced the development of the application. Conversion and installation ease are characteristics of the application. .

Operational Ease

Operational Ease describes the degree to which the application attends to operational aspects, such as start-up, back-up, and recovery processes.

Multiple Sites

Multiple Sites describes the degree to which the application has been developed for multiple locations and user organizations.

Facilitate Change

Facilitate Change describes the degree to which the application has been developed for easy modification of processing logic or data structure.

Step 3 Insert the TDI into the following equation to produce the value adjustment factor. VAF = (TDI * 0.01) + 0.65 , (i.e., TDI = 14 $\sum$ I=1)

Function Point Calculation*:*

Value Adjustment Factor, VAF= (TDI*0.01) + 0.65

Adjusted Development Project Functional Size (a DFP) = (UFP + CFP) *

VAF Where:

- DFP is the development project function point count
- UFP is the unadjusted function point count for the functions that will be available after installation

- CFP is the unadjusted function points added by the conversion unadjusted function point count
- VAF is the value adjustment factor (if the value adjustment factor was not calculated, VAF is 1.00)

*3. Initial Adjusted Application Functional Size (aAFP) = ADD * VAF*

Formula to calculate the application function point count:

Where:

- AFP is the initial application function point count.
- ADD is the unadjusted function point count of those functions that were installed by the development project.
- VAF is the value adjustment factor of the application. (If the VAF was not calculated, it will be assumed to be 1.00.)

4. Adjusted Enhancement Project Functional Size

This formula is to calculate the enhancement project function point count, aEFP = [(ADD + CHGA + CFP) * VAFA] + (DEL* VAFB)]

Where:

- EFP is the enhancement project function point count.
- ADD is the unadjusted function point count of those functions that were or will be added by the enhancement project.
- CHGA is the unadjusted function point count of those functions that were or will be modified by the enhancement project. This number reflects the size of the functions *after* the modifications.
- CFP is the function point count of those functions added by the conversion
- VAFA is the value adjustment factor of the application after the enhancement project is complete.
- DEL is the unadjusted function point count of those functions that were or will be deleted by the enhancement project.
- VAFB is the value adjustment factor of the application before the enhancement project begins

5. Adjusted Application Functional Size After Enhancement (aAFPA):

The formula to calculate the application function point count after an enhancement project, AFP = [(UFPB + ADD + CHGA) - (CHGB + DEL)] * VAFA

Where:

- AFP is the application's adjusted function point count.

- UFPB is the application's unadjusted function point count before the enhancement project begins.

Note: If this count is unavailable, it can be calculated using the formula UFBP = AFPB/VAFB; where AFPB is the adjusted application function point count before the enhancement project. VAFB is the value adjustment factor of the application before the enhancement project.

- ADD is the unadjusted function point count of those functions that were added by the enhancement project.
- CHGA is the unadjusted function point count of those functions that were changed by the enhancement project.
- CHGB is the unadjusted function point count of those functions that were changed by the enhancement project.
- DEL is the unadjusted function point count of those functions that were deleted by the enhancement project.
- VAFA is the value adjustment factor of the application after the enhancement project is complete.

Formulae for calculation, when the general system characteristics are not in use: 1. Development Project function size (DFP) = ADD+CFP

2. Initial application functional size (AFP) = ADD

3. Enhancement project functional size (EFP) = ADD+CHGA+AFP+DEL

4. Application function size after enhancement (AFPA). (AFPA)= (AFPB+ADD+CHGA)-(CHGB+DEL)

FP calculation with sample data*:*

Let us assume that, there is change request for a mainframe based application which includes the code changes in few impacted COBOL source modules, JCL changes and some DB2 table entry changes. For this typical WR, it is assumed that the total FP calculated using the complexity factor matrix value as 63.

Breakups, details for calculating the FP count:

For Internal Logic	3(S)*7(	21 FP
Files:	L):	counts
For External Interface		5 FP
Files:	1(S)*5(L	Counts

For External Inputs: 2(S)*3(L) + 2(M)*4(A): 14 FP Counts For EO: 1(S)*4(L) + 1(M)*5(A) + 2(C)*7(H): 23 Counts No external queries. And the total FP count is 63.

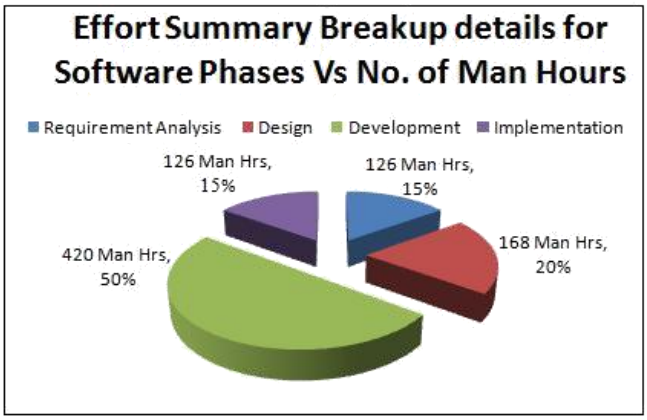
Whereas, Complexity levels - Simple (S), Medium (M), High (C), Predefined unadjusted FP type - Simple (L), Medium (A), Complex (H).

FP Calculation						
FP Items	Factor	Simple	Medium	Complex	Computation	
ILF		3	0	0	21	
EIF		1	0	0	5	
EI		2	2	0	14	
EO		1	1	2	23	
EQ		0	0	0	0	
Total FP					63	
Project Mgmt / Quality Control	10%				6	
Total Unadjusted FP					69	
VAF					0.96	
Total Unadjusted FP					67	
Contingency	5%				3	
Gross FP					70	
Total Man Hours	12	**Hrs / FP**			840	**Hours**
Total Man Days	8	**Hrs / Day**			105	**Days**
Total Man Months	22	**Days / Month**			5	**Months**

Based on the FP parameter assumptions of 12 hrs/FP for a typical mainframe application, it is estimated that, the total number of man hours required is 840 man hrs, which is around 105 days are required to complete this work request activity, which is roughly estimated to be around 5 months time of duration.

Why the software industries do does applies the FP method for estimation rather than other techniques [17]:

User requirements evolve rapidly in the early phases of a project. Decisions must be agreed upon by the users and the developer on which functions will be included in an application. Thus the, FPA is used as a tool [06]; To determine the size of a purchased application package by counting all the functions included in the package; To help users determine the benefit of an application package to their organization by counting functions that specifically match their requirements; To measure the units of a software product to support quality and productivity analysis; To estimate cost and resources required for software development and maintenance; It is also used as a normalization factor for the software comparison



Conclusion

Software engineering community has been showing keen interest in developing methodologies to improve the cost estimation process of the software project because of its profound impact on the management aspects[10]. In conclusion, Function point analysis has proven to be an accurate technique for sizing, documenting and communicating systems capabilities[06]. It has been a successfully used to evaluate the functionality of real time and

embedded code systems, such as robot based ware houses and avionics, as well as traditional data processing. As computing environments become increasingly complex, its proving to be a valuable tool that accurately reflects the systems we deliver and maintain [5]

References

1. A. J. Albrecht, "Measuring application development productivity." in Proc. IBM Applications Development Symposium., GUIDE Int. and SHARE Inc., IBM Corp., Monterey, CA, Oct. 14-17, 1979, p. 83.
2. A. J. Albrecht and J. E. Gaffney, "Software function, source lines of code and development effort prediction: A software science validation," IEEE Trans. Software Eng., vol. SE-9, no. 6, pp. 639-647.Nov. 1983.
3. A. J. Albrecht, "AD/M Productivity Measurement and Estimate Validation-Draft,'' IBM Corp. Information Systems and Administration, AD/M Improvement Program, Purchase, NY, May I . 1984
4. ANSI Standard Z94.2-1989. Industrial Engineering Terminology: Cost Engineering.
5. An introduction to Function Point Analysis, Rofer Heller, Vice president, A publication for information Technology Professional
6. Balasubramanian.M and Rajarajeswari.G, Software effort Estimation using Sizing method; International Journal of Fuzzy Systems**;** FS112011003; Print: ISSN 0974 – 9721 & Online: ISSN 0974 – 9608; 2011
7. Conference - September, 1997 Milne B.J., Luxford K.B.G. (ISBSG) - Worldwide Software Development, The Benchmark, Release 5 - March, 1998 - Chapters 5-6, pp. 23-36
8. Catherwood B., Sood M., AMS - Continued Experiences Measuring OO System Size - ESCOM
9. Function point analysis Quick reference card, CPM version 4.3
10. J. Frankvijay and C.Manoharan, Initial Hybrid Method for Analyzing Software Estimation, Benchmarking and Risk Assessment Using Design of Software, Journal of Computer Science 5 (10): 717-724, 2009
11. CHARLES R. SYMONS, Function Point Analysis: Difficulties and Improvements, IEEE transaction on software engineering, Vol:14, NO.I, January 1988,12.
12. Roberto Mali, Luca Santillo FUNCTION POINT ESTIMATION METHODS: A COMPARATIVE OVERVIEW, IEEE transaction on Computers
13. Hill P.R.(ISBSG) - Software Project Estimation, A Work book for Macro-Estimation of Software Development Effort and Duration - March, 1999 - Chapter 3
14. IFPUG - Function Point Counting Practices Manual, Rel. 4.0 - January, 1994