The Influence of Non Functional Characteristics in Effort Estimation using Function Point Analysis

Senthilkumar Murugesan and Chidambararajan Balasubramanian
Department of Computer Science and Engineering, Valliammai Engineering College, India

Abstract: *Software engineering is the discipline which paves the road map for the development of software within the given schedule and effort with the desired quality. The process begins with estimating the size, effort and time required for the development of the software and ends with the product. In most existing research on the effort estimation, the Function Point Analysis (FPA) method is used to estimate the effort, but not ensure the non- functional characteristics and quality factors of the project. In this paper, we study the uncertainty of effort estimation in the project and the impact of non functional characteristics in the effort estimation in detail. The refined model shows that the influence of non functional characteristics increases the accuracy of effort estimation in software project. By implications, the research suggests somewhat the impacts of non-functional characteristics in estimation is the most effective approach to improve the estimating accuracy may be to make estimators and developers more accountable in the software estimation.*

Keywords: *Software effort estimation, FPA, non functional characteristics, accuracy.*

Received November 26, 2013; accepted December 23, 2014

1. Introduction

Management in any projects starts with estimation. An effective estimation is the back bone of the development of any project. Without effective estimates, proper project planning and tracking is impossible [5]. If the estimates are too low then the project management tries to employ more personnel in order to expedite the development process that eventually results in the poor quality product and employee dissatisfaction [1].

The software engineering process begins with estimating the size, effort and time required for the development of the software and ends with their product and other work products built in different phases of development [3, 5]. One of the most common requirements in the software development projects is imposed by early estimation of the complexity and total cost [11]. To this end, various evaluation software metrics and methods are developed and one of the best known and widely accepted is the Function Point Analysis (FPA). Because of the method complexity, it is necessary to develop software which enables the automation of the whole process for calculating function point. In FPA the complexity weight metrics values of Function Point (FP) reflect the functional size of software. FPA was originally introduced by Albrecht to measure the size of the data processing system from the end users point of view, with the goal of estimating the development effort [10].

As the system grows in size, it is really difficult to estimate the size of the software earlier in the development. FPA breaks the system into smaller pieces so that intricacies of the systems become more visible and can be analyzed better [9]. FPA measures size of the software on the basis of the functionalities to be provided by the software. FPA estimates the size of software in terms of Function Point Counts (FPC) which can be converted into Size Line Of Code (SLOC) easily if the equivalent SLOC for unit FPC is available [12].

2. Function Point Analysis

Conceptually FPA defines data in two levels; data at motion and data at rest. Every application has numerous elementary processes which includes various transactions for data movement. It includes transactions bringing data into the application domain and transactions taking data out of the application domain. These are referred to as transaction functions [13]. The data maintained by the application or by another application are known as data at rest and referred to as data functions [14].

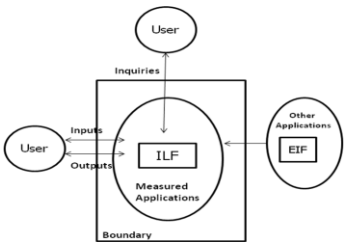

Figure 1. Types of data and transaction functions.

2.1. External Inputs (EI)

EI is the process in which data comes from outside of the application domain. The data may come from the input screen or from the other application. Control and business data both are counted as EI. The input can manipulate one or more files maintained by the

application. If an input is performing insertion, updating and deletion then it is counted as three EIs.

2.2. External Outputs (EO)

The process in which any derived data crosses the boundary of application from inside to outside is known as EO. Derived data here means the processed data and not the data through simple retrieval from the external interface files or internal logical files. It usually is the result of some calculation or algorithmic operation [6].

2.3. External Inquiry (EQ)

The process with both input and output components retrieves data either from the internal logical files or from the External Interface Files (EIF). EQ does not update any internal logical files or external interface files [14].

2.4. External Interface Files (EIF)

User identified logically related data stored outside the application boundary is known as EIF. The file containing the logically related data can be counted as external interface files or internal logical files but not both. Each EIF should have at least one EI or EO for it [13].

2.5. Internal Logical Files (ILF)

User identified logically related data maintained inside the application through EI is known as ILF. These files should have at least one EI for it.

2.6. File Type Referenced (FTR), Data Element Type (DET) and Record Element Type (RET)

FTR is a file reference by any transaction. It should be either an internal logical file or external interface file. DET is unique information in FTR. DET could be information for the instigation of any information or could be additional information about the transaction. RET is a unique sub group of data in FTR. DET, RET and FTR are used in the calculation of number of EI, EO, EQ, EIF and ILF. Dependencies of EI, EO, EQ, EIF and ILF on DET, RET and FTR are shown in the following table.

Table 1. Dependencies of factors.

	DET	RET	FTR
EI	√		√
EO	√		√
EQ	√		√
ILF	√	√	
EIF	√	√	

2.6.1. Steps for Counting FPs

Required information for counting is obtained from the software requirement specification. The steps for counting FPs are as following [3, 5, 14]:

1. To identify data functions (EIF and ILF) and rate them.
2. To identify transaction functions (EI, EO and EQ) and determine their complexity.
3. To compute unadjusted FPs. Number of EI, EO, EQ, ILF and EIF for each complexity level (simple, average and nominal) is obtained and the corresponding weight for each complexity level is multiplied with the count to finally get the unadjusted FPC.
4. To determine the ratings of 14 general system characteristics.
5. To calculate Value Adjustment Factor (VAF).

$$VAF = (TDI * 0.01)+0.65 \quad (1)$$

Where TDI=total degree of influence obtained by multiplying the ratings of general system characteristics.

6. To compute FPC.

$$FPC = UFP*VAF \quad (2)$$

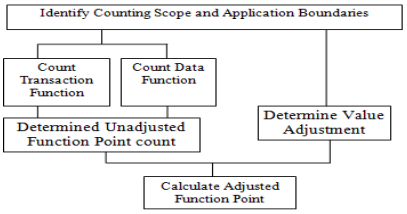

Figure 2. Function Point Counts.

2.7. System Objectives and Current Functionalities

The current scenario includes a beginning estimate of how many hours a project team requires to complete the software project [14]. This estimate is broken down by 4 phases that the software project will go through in its life cycle. The estimator may enter the projected project start date, number of staff, and the average weekly hours worked by each staff member to view an estimated project completion date [14]. Additionally, the estimator may enter rates of pay and known project expenses to see estimates of the project's costs. These estimates should not be used as the final project estimates. They can, however, help to guide the estimator in the early planning phases of the project to give an idea of a reasonable number of hours to allocate for the project. These estimates do not guarantee that the project team will finish the project within the hours provided in the estimates. Additionally, all estimation of software projects should be integrated into a thoughtful risk management program. The main point to remember

regarding the interface of estimates with project risks is that if any of the known risks materializes, the estimates should be adjusted to add time to the schedule to recover from the damage suffered by the materialization of the risks.

2.7.1. Current Methods and Procedures

The current system considers all functional aspects of a software component to determine the total work effort and the total resource hours with the duration in days and weeks. A detailed estimation includes total work effort and resource allocation. These estimations involve factors such as specification information, interfaces involved, reports and correspondences, infrastructure services and resources, work effort resources and management consulting services and resources. These factors are carefully analyzed and its functional characteristics are captured to estimate the total work effort [7].

First of all, the general details of a project like client name, project name, project start date, project type, PRCP version, project start date, framework name, method and maturity are recorded. Then, all the instances of specifications, interfaces, reports and correspondences are captured based on its functional values. Then, a total of effort by phase is calculated by summing up all the above characteristics [8]. Then, risk contingency is captured by recording the percentage of risk associated with the project and the total risk is evaluated by multiplying the risk percentage with the effort by phase totals. Finally, the effort and contingency is calculated by summing up the risk contingency and the effort by phase totals. A diagrammatic representation of the above mentioned procedure is given below.

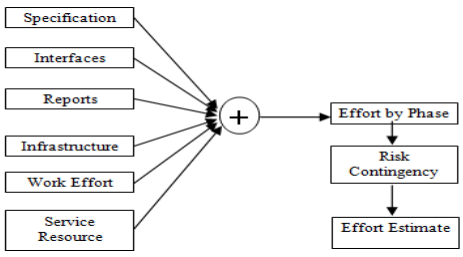

Figure 3. Estimation overview.

2.7.2. FP Calculation

Formula for calculating the FP count:

1. Function Count (FC):

$$FC=\sum_{i=1}^{5}\sum_{J=1}^{3} X_i W_i. \quad (3)$$

2. FP:

$$FP=FC*[0.65+0.01(\sum_{k=1}^{14} C_k)]. \quad (4)$$

Where X_i = function, W_j = weight$_j$, C_k = complexity Factor *k*.

3. Converting FP to SLOC.

$$SLOC =16 * [(SLOC)/FP] \quad (5)$$

4. Calculating effort.

$$Effort=EAF*A*(SLOC)^{EX} \quad (6)$$
$$EAF=CPLX * TOOL$$

Where A=3.2 and EX=0.38 (based on development mode), *CPLX* = 1.3 (based on development language), *TOOL* = 1.1 (based on development tool).

3. Impact of the Non-Functional Characteristics over the FP Analysis

The proposed methodology works by considering the non-functional characteristics in effort estimation. The non-functional characteristics can be defined as follows: In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. This should be contrasted with functional requirements that define specific behaviour or functions. For ease of use, the requirements can be thought of as belonging to a type [12].

Functional requirements are the fundamental or essential subject matter of the product. They describe what the product has to do or what processing actions it is to take [3].

Non-functional requirements are the properties that the functions must have, such as performance and usability. These requirements are as important as the functional requirements for the product's success.

Several non-functional characteristics are involved in building a software product. Those can be combined and grouped under the following types:

- Look and feel requirements.
- Usability and humanity requirements.
- Performance requirements.
- Operational and environmental requirements.
- Maintainability and support requirements.
- Security requirements.
- Cultural and political requirements.
- Legal requirements.

Though, all these features influence the success of software product, the performance criteria plays a vital role among these characteristics. Hence, our estimation technique takes the performance criteria into consideration. The performance criterion only directly affects the effort estimation in the non-functional requirements [7]. If the performance characteristics are included in the effort estimation, obviously it provides better effort estimation of the project. So, in this paper we are focusing only on the performance characteristics. With this intent, let us describe the performance characteristics and the 8 of its inherent features.

3.1. Performance

Computer performance is characterized by the amount of useful work accomplished by a computer system compared to the time and resources used. The performance of any computer system can be evaluated in measurable, technical terms, using one or more of the following metrics [7, 13].

- Speed and latency.
- Safety critical.
- Precision or accuracy.
- Reliability and availability.
- Robustness or fault-tolerance.
- Capacity.
- Scalability or extensibility.
- Longevity.

3.1.1. Speed and Latency

Speed and latency specifies the amount of time available to complete specified tasks. These requirements often refer to response times. They can also refer to the product's ability to operate at a speed suitable for the intended environment. Some product, usually real-time products, must be able to perform some of their functionality within a given time slot. Failure to do so may mean catastrophic failure (e.g., a ground-sensing radar in an airplane fails to detect an upcoming mountain) or the product will not cope with the required volume of use (e.g., an automated ticket-selling machine).

3.1.2. Safety

Safety criteria refer to quantification of the perceived risk of damage to people, property, and environment. Safety can also be defined to be the control of recognized hazards to achieve an acceptable level of risk. This can take the form of being protected from the event or from exposure to something that causes health or economical losses. It can include protection of people or of possessions. The main motivation behind safety is to understand and highlight the damage that could potentially occur when using the product within the expected operational environment.

3.1.3. Precision or Accuracy

Precision or accuracy refers to the quantification of the desired accuracy of the results produced by the product. The main motivation behind accuracy is to set the client's and users' expectations for the precision of the product.

3.1.4. Reliability and Availability

The reliability is usually expressed as the allowable time between failures, or the total allowable failure rate. In reliability theory and reliability engineering, the term availability can be defined as the degree to which a system, subsystem or equipment is in a specified operable and committable state at the start of a mission, when the mission is called for at an unknown, i.e., a random, time. Simply put, availability is the proportion of time a system is in a functioning condition. This is often described as a mission capable rate. Mathematically it can be stated as the ratio of the total time. A functional unit is capable of being used during a given interval to the length of the interval.

3.1.5. Robustness or Fault Tolerance

The term robustness is the ability of a computer system to cope with errors during execution or the ability of an algorithm to continue to operate despite abnormalities in input, calculations, etc. The harder it is to create an error of any type or form that the computer cannot handle safely, the more robust the software is. Robustness specifies the ability of the product to continue to function under abnormal circumstances.

3.1.6. Capacity

Capacity specifies the volumes that the product must be able to deal with and the amount of data stored by the product. The main motivation behind this is to ensure that the product is capable of processing the expected volumes.

3.1.7. Scalability or Extensibility

Scalability specifies the expected increase in size that the product must be able to handle. As business grows, the software products must increases its capacity to cope with the new volumes. The main motivation is to ensure that the designers allow for future capacities.

3.1.8. Longevity

Longevity specifies the expected lifetime of the product. The main motivation is to ensure that the product is built based on an understanding of expected return on investment.

4. Evaluation Process

This section deals with the evaluation of FPs and analyzes the performance.

4.1. Implementation

The proposed model is used to analyze the impact of non functional characteristics in the FP analysis for effort estimation. For implementing the impact of non functional characteristics in the FPA, this work selects FPA as the base and calculate the correctness factor of non-functional factors. Finally, the calculated correctness factor is added with the FPC to make as the high accurate effort estimation. The implementation is performed for the data given by the

brain lead software solutions, Chennai. For evaluation purpose, the model is developed in Visual Basic to calculate FP and correctness factor of non-functional characteristics. The estimated efforts are compared by using MS Excel and bar chart is developed.

4.1.1. Implementation Description

With the existing functional point counting methodology, the non functional characteristics that are above discussed can be added. Following are the steps to be followed in order to perform this.

List all the 8 performance characteristics with its description.

Specify the value for each factor. The value ranges from 0 through 5 depending upon the factor's influence on the product to be developed.

Specify the complexity for each factor. The complexities vary between low, medium and high. The following table specifies the scaling table for these complexity factors.

Table 2. Complexity factors.

Complexity	Scaling
Low	5
Medium	10
High	20

Multiply each value of the performance factor with the complexity and sum up all those 8 values.

Its percentage value gives the overall correctness factor which will be added to the FP estimate to find the corrected estimated value.

4.1.2. Adjusted Function Point Calculation

The formula to determine the correctness factor and adjusted FP are given below:

$$CF = 0.01 * (\sum_{i=1}^{8} v_i * C_i) \quad (7)$$

Where V_i=value of each performance factor and C_i=complexity factor.

$$AFP = FPC + CF \quad (8)$$

Where *AFP*=adjusted FP and *CF*=correctness factor.

4.2. System Frameworks

The framework of how the above mentioned formula can be incorporated in the existing FP analysis can be inferred from the following diagram.

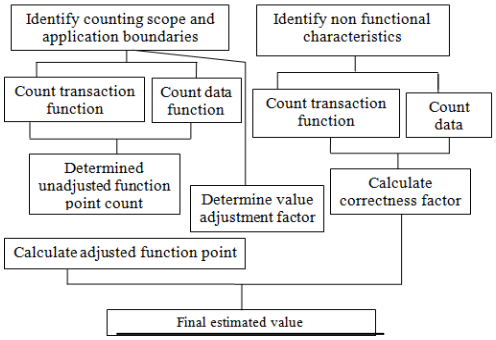

Figure 4. System framework.

This system framework of final effort estimation deals both unadjusted FP count and correctness factor of the non-functional characteristics in the worktable calculation.

4.3. Worktable Calculation

We have calculated the FP counts for the collected data of the given project and the non-functional characteristics are also calculated using Visual Basic forms. The final worktable of the correctness table is given below.

Table 3. Correctness factor calculation.

No.	Non-Functional characteristics	Value	Complexity
1	Speed and latency.	4	Medium
2	Safety critical.	3	Medium
3	precision or accuracy.	3	Medium
4	Robustness or fault-tolerance.	5	Medium
5	Capacity.	1	Medium
6	Scalability or extensibility.	4	Low
7	Longevity.	2	Low
8	Reliability and availability.	3	Low
	Correctness factor.	**2.05**	

The general system characteristics are recorded and the value adjustment factor is computed. The work sheet of this is shown below:

Table 4. Value Adjustment Factor Calculation (VAT) calculation.

No.	General System Characteristic	Value
1	Data communications	4
2	Distributed data processing	4
3	Performance	3
4	Heavily used configuration	3
5	Transaction rate	4
6	On-line data entry	5
7	End-user efficiency	2
8	On-line update	5
9	Complex processing	3
10	Reusability	4
11	Installation ease	3
12	Operational ease	3
13	Multiple sites	4
14	Facilitate change	2
	Value adjustment factor	**1.14**

The final estimation worksheet is shown below:

Table 5. AFP calculation.

Name	DET	FTR/RET	Rating	Value	Multiplier	Total Value
Any EI	2	1	L	3	1	3
Any EI Name	6	4	H	6	1	6
Any EO Name	1	4	L	7	1	7
Any ILF Name	2	4	L	7	1	7
Any EIF Name	3	4	L	5	1	5
Any EI with Add, Change and Delete	4	4	A	4	3	12
Another EO	5	4	A	5	1	5
Total Unadjusted FPs						45
Value Adjustment Factor						**1.14**
Correctness Factor						**2.05**
Total Adjusted FPs						**53.35**

The final total adjusted FP value (53.35) obtained will be more accurate than the existing estimation model (51.30). Hence, this type of estimation yields more accurate results which will help the project manager to accurately plan the budgeting and scheduling of tasks.

4.4. Performance Evaluation

The proposed estimation model is compared to the existing method. The results are analyzed and it is found that the proposed model yields a result which is more accurate than the existing method. The effort estimation using FP for the given project is less than the proposed method with non-functional characteristics influences, but the accuracy of the effort estimation is higher than the existing model. Since, the software projects need more accuracy in the estimation process, because and if it is inaccurate, simultaneously the project results in poor estimation leading to the failure of the project.

The following graph shows the comparative results:

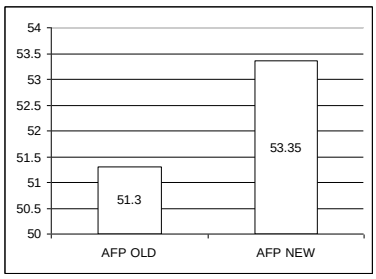

Figure 5. Comparison of AFP old with AFP new.

Where AFP old=estimated value without non-functional factors and AFP new=estimated value with non-functional factors.

5. Conclusions and Future Works

This work proposes the effort estimate using the FP's with the influences on non-functional characteristics which increase the accuracy and quality leading to the effective estimation of the software projects based on our results and comparison chart. This work is very useful to the time critical and mission critical project which needs more accuracy and high quality estimation. Moreover, the only limitation of this work is high cost than the existing method for estimation [8, 14]. So this may not be suitable to smaller projects with small budgets and time frame.

Future work will utilize these adjusted FP analysis in the planning and scheduling phases of the project and make it a hybrid tool which provides all the phase of software estimation in a single tool.

References

[1] Azath H. and Wahindabanu R., "Efficient Effort Estimation System viz Functionn Point and Quality Assurance Coverage," *IET Software*, vol. 6, no. 2, pp. 335-341, 2012.

[2] Berander P. and Jonsson P., "Hierarchical Cumulative Voting-Prioritization of Requirements in Hierarchies," *International Journal of Software Engineering and Knowledge Engineering*, vol. 16, no. 6, pp. 819-849, 2006.

[3] Bouckkif S., Ahamed F., Serhani A., and Khalil I., "Integrating Function Point Project Information for Improving the Accuracy of Effort Estimation," *in Proceeding of International conference on Advanced Engineering Computing and Applications in Science*, Valencia, pp. 193-198, 2008.

[4] Fetcke T., "The Warehouse Software Portfolio-A Case Study in Functional Size Measurement," *Technical Report*, 1999.

[5] Hu H., Yang B., and Jia L., "A Study of Uncertainty in Software Cost and its Impact on Optimal Software Release Time," *IEEE Transaction on Software Engineering*, vol. 34, no. 6, pp. 813-825, 2008.

[6] Lavazza L., "A Systematic Approach to the Analysis of Function Point-Cosmic Convertibility," *International Workshop on Software Measurement*, pp. 10-12, 2010.

[7] Lavazza L., Bianco V., and Garavaglia C., "Model Based Functional Size Measurement," *in Proceeding of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, Kaiserslautern, pp. 100-109, 2008.

[8] Lavazza L., Bianco V., and Liu G., "Analytical Convertibility of Functional Size Measures: A Tool Based Approach," *in Proceeding of International Conference on Software Process and Product Measurement*, Assisi, pp. 160-169, 2012.

[9] Matijevic T., Ognjanovic I., and Sendeji R., "Enhancement of Software Projects' Function Point Analysis Based on Conditional non-Functional Judgements," *in Proceeding of*

Mediterranean Conference on Embedded Computing, Montenegro, pp. 250-254, 2012.

[10] Kaushal P., "Software Effort Estimation and Risk Analysis-A Survey," *International Journal of Engineering and Innovative Technology*, vol. 1, no. 1, pp. 18-22, 2012.

[11] Rajan B. and Kumar M., "Impact of Performance Metrics in Software Effort Estimation using Function Point Analysis," *Information-An International Interdisciplinary Journal*, pp. 2255-2268, 2014.

[12] Stirling W., Frost R., Nokleby M., and Luo Y., "Multicriterion Decision Making with Dependent Preferences," *in Proceeding of IEEE Symposium on Computational Intelligence in Multi Criteria Decision Making*, pp. 227-234, 2007.

[13] Soltani S., Asadi M., Hatala M., Gašević D., and Bagheri E., "Automated Planning for Feature Model Configuration Based on Stake Holders Business Concerns," *in Proceeding of IEEE /ACM International Conference on Automated Software Engineering*, Lawrence, pp. 536-539, 2011.

[14] Yu Z., Zhou X., Nakamura Y., and Yu Z., "Toward an Understanding of User Defined Conditional Preferences," *in Proceeding of IEEE International Conference on Dependable Automatic and Secure Computing*, pp. 203-208, 2009.

Senthilkumar Murugesan received the B.E Degree in Computer Science and Engineering from Madurai Kamaraj University, in 2003.He received the M.E Degree in Computer Science and Engineering from Anna University, in 2006. He received his Ph.D. Degree from Anna University, in 2015. Currently, he is an Associate Professor in CSE Department at Valliammai Engineering College, Chennai. His interests are in Software Cost Estimation and Wireless Communication. He published more 25 research papers in Reputed International Journals.

Chidambararajan Balasubramanian received the M.E. degree in Microwave and Optical engineering from the Kamaraj University. He received the Ph.D. degree in Information and Communication engineering from the Anna University. Currently, He is a professor/Principal at Valliammai Engineering College, Affiliated to Anna University, Chennai. He has 20 years of teaching experience in government and reputed private institutions. He is a member of professional societies like IEEE, IETE, IEI, ISTE, CSI, ISOI, etc. He has published several technical papers in national and international journals and conferences His research interests include Software Engineering and Networking.