GUI of esrcTool: A Tool to Estimate the Software Risk and Cost

Mohd. Sadiq, Sunil, Sherin Zafar
Section of Computer Engineering, Department of University Polytechnic, Faculty of Engineering and Technology, Jamia Millia Islamia
(A Central University), New Delhi-110025, India

Mohammad Asim, R. Suman
M. Tech. Scholar, Department of Computer Science and Engineering, Al-Falah School of Engineering and Technology, Faridabad;
Maharshi Dayanand University, Haryana, India.

Abstract——**Function Point Analysis was developed first by Allan J. Albrecht in the mid 1970s. It was an attempt to overcome difficulties associated with lines of code as a measure of software size, and to assist in developing a mechanism to predict effort associated with software development. Function Point is a well known established method to estimate the size of software projects. There are several areas of the software engineering in which we can use the function point analysis (FPA) like project planning, project construction, software implementation etc. In this paper we have implemented the architecture of the esrcTool in C Language. This tool is used for two different purposes, firstly, to estimate the risk in the software and secondly to estimate the cost of the software. The esrcTool is based on SRAEM i.e. Software Risk Assessment and Estimation Model, because in this model FP is used as an input variable, and in order to determine the cost of the software we have used the International Software Benchmarking Standards Group Release Report (ISBSG).**

Keywords-Function Point, Software risk, Cost, SRAEM, esrcTool.

1. INTRODUCTION

At the beginning of the 1970s, researchers at IBM initiated studies aimed at determining what critical variables were involved in programming productivity. Instead of considering code volume or complexity, they discovered that a system would be better evaluated by analyzing the functions executed by programs and mapping pertinent questions to estimating and evaluating software's development productivity in heterogeneous environments (Albrecht, 1979). Function Point is a well known established method to estimate the size of software system and software projects. Originally, the method was used in the early phases of the waterfall model such that the implementation effort could be estimated on the basis of input and output behavior as defined in the functional documentation. As the size and the complexity of software increases, it becomes increasingly important to develop high quality software cost effectively within a specified period. In order to achieve this goal, the entire software development processes need to be managed based on an effective plan. The subjects of estimation in the area of software development are size, effort invested, development time, technology used and quality [8, 9, and 19].

978-1-4244-5586-7/10/$26.00 © 2010 IEEE

Function Point is a measure of software size that uses logical functional terms, business owners and users, more readily understand. Albrecht's model of functional specifications requires the identification of five types of components, namely input, output, inquiry elementary processes, logical internal elementary processes, and logical internal and external interface files. The actual calculation process itself is accomplished in three stages: (I) determine the unadjusted function point (UFP), (II) Value adjustment factor (VAF) and (III) Adjusted function points (AFP). For detailed calculations of the FP, readers are advised, please refer to [20, 21].The paper is organized as follows. In section 2 we present the background and related work. In section 3, we have explained the esrcTool. Experimental work is carried out in section 4. Finally we conclude the paper in section 5.

2. BACKGROUND AND RELATED WORK

Researchers, scientist and academician in the field of software engineering have developed a lot of models/ tools according to their need and requirements. In [3] authors have developed a SRAEM (Software Risk Assessment and Estimation Model). Using this model it is possible to predict the possible results of software projects with good accuracy. SRAEM model not only assess the risk but it also estimate the risk. In this model the risk is estimated using risk exposure and software metrics of risk management and this metric is based on Mission Critical Requirements Stability Risk Metrics (MCRSRM) [6]. This software metrics is used when there are changes in requirements such as addition, subtraction, or deletion. The proposed architecture gives the incremental risk for every phase and also the total cumulative risk as the software progress from phase to phase. In [4] the authors have developed a Software Estimation Tool Based on Software Engineering Metrics Model. But in this tool there is no description regarding the costing of the software using ISBSG. In this paper we have developed the architecture of the esrcTool in order to estimate the risk and cost of software. The snapshot of the tool that the authors of [4] have developed is given in figure 01. There are a few published models that evaluate the risk of software projects. In [25] the authors have developed a model and prototype tool to manage software risks. Soft Risk prototype is a tool prototype to manage software development risks. Java language has been chosen as development language of the

prototype. Another model named Software Engineering Risk Model (SERIM) focuses on three risk elements: (i) technical risk, (ii) cost risk, and (iii) schedule risk.

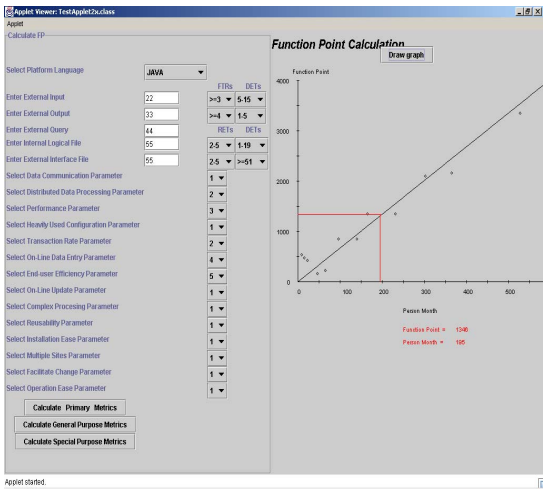


Figure 1.

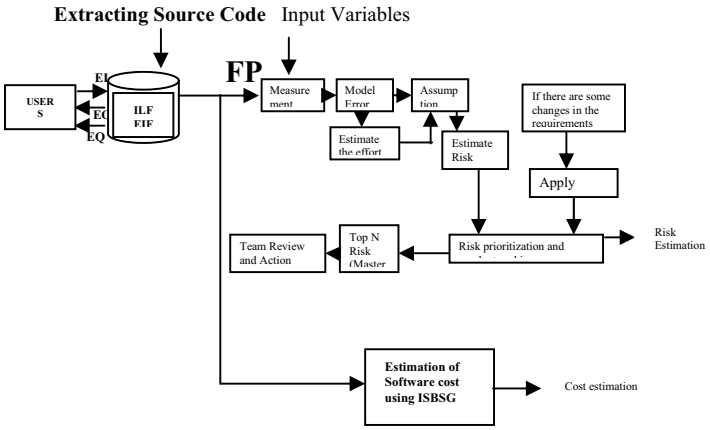
This model does not take into account of the software complexity issues, which plays an important role in determining the risk for the software projects. It also does not account for issues related to requirements. The proposed architecture addresses this problem and estimate the risk at each phase of software as it progress from phase to phase. In the series of risk assessment models there is another model called Software Risk Assessment Model (SRAM)[17]. This model considers the nine critical risk elements (i) complexity of the software; (ii) staff involved in the projects (iii) targeted reliability (iv) product requirement (v) method of estimation (vi) method of monitoring (vii) development process adopted (viii) Usability of software (ix) tools. The above existing risk assessment models does not include the sources of estimate uncertainty, i.e. measurement error, model error and assumption errors. Existing methods have considered the prioritization as a single step of risk assessment but does not specify how prioritization would be done. In [26] the authors have proposed the architecture of an esrcTool to estimate the software risk and Cost. In the continuation of this work we have implemented the esrcTool in C Language. The Snapshot of the GUI of the esrcTool is given in section 4.

3. EXPLANATION OF THE ESRCTOOL

Initially the proposed tools i.e. esrcTool [26] extract the source code of the program/ software. We have implemented this tool in C language in order o estimate the risk of the software and also to find out the cost of software. The first step of this tool is to calculate the function point of the software. The value of the function point would be used as an input to the measurement error, model error, and assumption error. The architecture of the proposed tool [26] is given in figure-01

3.1 Estimation of the Risk

There are 3 dimensions of software risk i.e. technical risk, organization and environmental risk [18]. Each software models have some weaknesses and also have some advantages [10], and a technical report on the software risk evaluation method is available at [16, 26].



Architecture of the proposed esrcTool

(Figure 2. Adopted From [26])

3.1.1 Measurement Error

This error occurs if some of the input variables in a model have inherent accuracy limitations. As a result of Chris F. Kemerer [1], function points are assumed to be at least 12% inaccurate. Thus if we estimate a product size of 1000 function points, measurement error could mean that the real size is anywhere between 880 and 1120.

3.1.2 Model Error

Factors that affect error but are not included explicitly in the model contribute to the model error. For example such as 0.5 person-days per function point is usually obtained from results observed for recalled from previous projects. It is unlikely that any future projects will achieve the same ratio, but the model is expected to all right on average. If you base a model on past project data, you should calculate the associated inaccuracy by using the mean magnitude relative error. Thus if you have an estimation model with an inherent 20% inaccuracy and your product is 1000 function points in size, your estimate is likely to be between 400 and 600 person days.

3.1.3 Assumption Error

This occurs when we make incorrect assumptions about a model's input parameters. For example, your assessment that a product size is 1300 function point rests on the assumption that you have correctly identified all the customer requirements. If you can identify your assumptions, you can investigate the effect of their being

invalid by assessing both the probability that an assumption is incorrect and the resulting impact on the estimate. This is the form of risk analysis. For example you believe that there is a 0.3 probability that the requirement complexity has been underestimated and, if it has, you estimate another 390 function point. At this point the concept of risk exposure is used to calculate the effective current cost of a risk and can be used to prioritize risk that requires countermeasure. Mathematically it can be written as Probability of risk occurring * Total loss if risk occur. These three errors are used to estimate the risk exposure. We have also included the MCRSRM in the proposed tool. In MCRSRM, the risk is computed when there are some changes in the requirements (addition, modification, or deletion). So total risk can be computed as [b/a] i= 1 to n +K [A [c/d] i +B [d/b]i +G [e/b] i] .Where [b/a] i= (Number of mission critical requirements)/ (Total number of requirements) at the input of phase number i. Ki is the penalty for adding, modifying or deleting of requirements during phase number i.[17]. The above information was related with the estimation of the risk in the software. In the next section we have discussed the cost estimation of the software using ISBSG.

3.2 Cost Estimation of the Software

In the proposed tool we have used the International Software Benchmarking Standards Group (ISBSG). It is an international Group of representatives from international metrics organizations who collect project data from countries like, India, Hong Kong Germany, Japan, and USA. ISBSG Release 6 Report provides the cost value for the software projects. Cost data is derived from 56 projects representing a broad cross section of the software industry. After going through these software projects, the ISBSG conclude that median cost to develop a function point is $US 716, and the average cost is $ US 849 per function point. For more information about the ISBSG please visit: www.ISBSG.org.au

In the calculation of the FP, calculating the value adjustment factor (VAF) is an earmark of the general functionality provided to the user. The VAF is derived from the sum of the degree of influence (DI) of the 14 general system characteristics (GSCs). The DI of each one of these characteristics ranges form 0 to 5 as follows:

(i) 0 – no influence;
(ii) 1 – incidental influence;
(iii) 2 – moderate influence;
(iv) 3 – average influence;
(v) 4 – significant influence; and
(vi) 5 – strong influence.

The general characteristics of a system are : (i) data communications; (ii) distributed data processing; (iii) performance; (iv) heavily used configuration; (v) transaction rate;(vi) online data entry; (vii) end user efficiency (viii) online update(ix) complex processing; (x) reusability; (xi) installation ease; (xii) operational ease; (xiii) multiple sites; (xiv) facilitate change. The third and the last stage is the final calculation of the function points. With the help of the following equation we can get the total points of an application.

AFP = UFP * VAF. Where AFP = adjusted function points; UFP = unadjusted function points; and VAF = value adjustment factor. [1, 5, 8, 9]

Function points are computed by completing the table 1. Five information domain characteristics are determined and counts are provided in appropriate table location [19]. In table 1 the MP is the measuring parameters, i.e. External Input (EI), External Output (EO), External Query (EQ), Internal Logical File (ILF), External Interface File (EIF) and UFP is the unadjusted function point.

TABLE 1

MP	Count	Simple	Average	Complex	Result
EI	X	3	4	6	
EO	X	4	5	7	
EQ	X	3	4	6	
ILF	X	7	10	15	
EIF	X	5	7	10	
UFP	**TOTAL**				

Once these data have been collected, a complexity value is associated with each count. Organization that use FP methods develop criteria for determining whether a particular entity is simple, average, or complex. To compute the AFP the following relationship is used [13, 14, 15].

$$AFP= UFP*[0.65+0.01*\sum (f i)] \quad (1)$$

4. EXPERIMENTAL WORK

In this section we have presented how the proposed esrcTool would be useful to estimate the risk in software and also to estimate the cost of software? In this paper we have considered the projects developed by the students of Master of Technology (M.Tech.) of Computer Science and Engineering. The project that we have considered is “A Mini Software for Numerical Integration (MSNI)”. This Software basically calculates the values of the given Integrand after applying some proposed algorithm and all the existing algorithms of the numerical computations like Simpson’s 1/3 rule, Simpson’s 3/8 rule and so on. In MSNI we have collected 21 different requirements; we have shown the prioritization of these requirements in the Figure- 04. For more detail about the Software requirements elicitation and prioritization, please refer to [2, 5, 7, 11, and 12]. The GUI of the proposed tool is given in figure – 3.

In order to estimate the function point of the MSNI we have used the esrcTool and as well as the tool proposed by [4] . The value of function point of MSNI found to be 50. According to the measurement error the actual size of the function point varies from 44-56. In order to find out the model error we have assumed that the 0.2 person –days per function point. No estimation model can include all factors

that affect the effort required to produce a software product. Suppose we have an estimation model with an inherent 20 % inaccuracy and our product is of 50 function points in size, our estimation is likely to be between 8-12 person days.

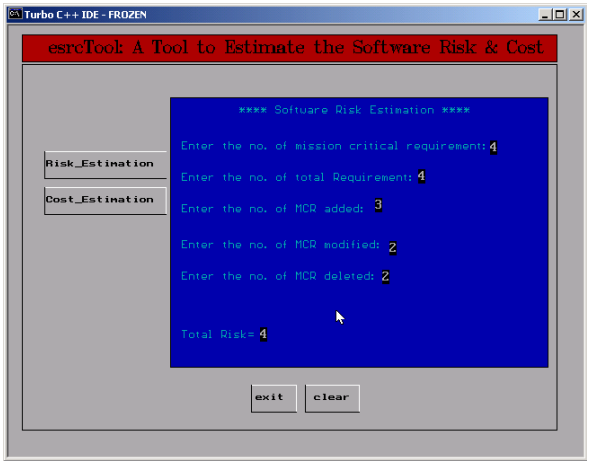


Figure 3 (Snapshot of the GUI of the esrcTool)

The assumption error occurs when we have some incorrect assumption about the models input parameter. Our assumption is that the product size is of 50 function points rest on the assumption that we have correctly identified all the requirements. If we can identify our assumption, we can investigate the effect of their invalid by assessing both the probability that an assumption is incorrect and the resulting impact on the estimate. This is the called the risk analysis. If we assume that there is 0.4 probabilities that the requirement complexity has been underestimated, so we estimate another 2 function point. We can estimate the risk exposure from the following formula: Risk Exposure = E2-E1, where E1 is the effort if the original assumption is true, and E2 is the effort if the alternative assumption is true and P2 is the probability that the alternative assumption is true. So in our case E1=10 person days, E2= 50+2*0.2= 10.4=11(approximately). The risk exposure =11-10=1 person days. In this paper we have considered three projects designed and developed by our students and these projects are implemented in C and C++ language. The values of the function point of theses projects are 50, 150, and 150 respectively. The results for the given software that we have got from the proposed esrcTool are summarized in table 2.

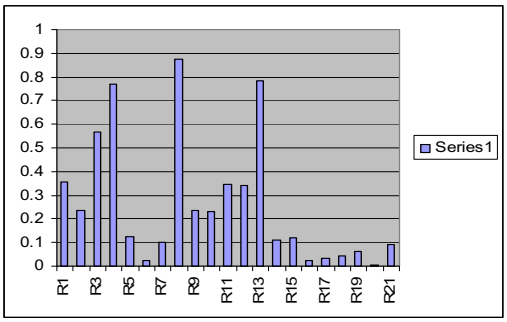


Figure 4.

TABLE 2 RESULTS FROM ESRCTOOL [26]

Project with FP	MeE	MoE	AsE	RE	Cost
Project 1 FP=50	44 FP to 56 FP	8-12 person days	Addition of 2 more FP	1 person days	44*cost of one FP – 56* cost of one FP
Project 2 FP=100	88 FP to 122 FP	16-24 Person days	Addition of 4 more FP	11 persons days	88*cost of one FP – 122 * cost of one FP
Project 3 FP=150	72 FP to 228 FP	38-42 Persons days	Addition of 60 more FP	42 persons days	72*cost of one FP – 228 * cost of one FP

5. CONCLUSION AND FUTURE WORK:

In this paper we have implemented the esrcTool using C language. From the proposed tool it is easy to estimate the risk in the software and also to estimate the cost of the software. The cost of the software depends on the value of the function point. In this paper we have applied the function point approach as an input parameter into the esrcTool. In this paper we have applied the proposed tool on the MSNI i.e. Mini Software for Numerical Integration and also two other projects that are based on Software Engineering and Computer Graphics. From the proposed tool it is easy to find out the cost of software and estimate the risk of those software's projects that were designed and developed by the graduate and post graduate students. In future we will elicit the software requirements after adding the threat in to it and then we will prioritize it using analytic hierarchy process and quality function deployment, and after this we will

generate the results of that software using the proposed esrcTool.

ACKNOWLEDGEMENT:

The authors would like to thank *Mr. Iqbal Azam*, Principal, University Polytechnic, Faculty of Engineering and Technology, Jamia Millia Islamia (A Central University), New Delhi-25, India ; and *Mr. Jawad Ahmad Siddiqui*, Chairman, Al-Falah School of Engineering and Technology, Dhauj, Faridabad, Haryana, India, for his valuable support , guidance and encouragement

REFERENCES:

[1] Chris F. Kermerer, "Reliability of Function Points Measurements, A Field Experiment, Communication of the ACM, pp. 85- 97, Vol.36, February 1993.

[2] D. Firesmith, " Prioritizing requirements", Journal of Object Technology, Volume 3, No.8, September 2004

[3] Daya Gupta, Mohd Sadiq, "Software Risk Assessment and Estimation Model", International Conference on Computer Science and Information Technology, IEEE Computer Society, Singapore, 2008. pp 963-967

[4] Daya Gupta, SatyaPal Jee Kaushal, Mohd. Sadiq, "Software Estimation tool Based on Software Engineering Metrics Model", IEEE International Conference on Management of Innovation & Technology, Bangkok, Thailand. Pp.623-628.

[5] J. Karlsson, "Software Requirements Prioritizing", Proceedings of the International Conference on Requirement Engineering, 1996.

[6] Joseph S. Sherif, "Metrics for Software Risk Management", ISMN#0-7803-3274-1, pp.507-513.

[7] LI Zong-yong, WANG Zhi-xue, YANG-ying, WU Yue, LIU Ying, " Towards multiple ontology Framework for Requirements Elicitation and Reuse", 31st IEEE Annual International Computer Software and Application Conference, 2007.

[8] Mohd. Sadiq, and Shabbir Ahmed, "Computation of Function Point of Software on the basis of Average Complexity", Proceedings of 2nd International Conference on Advanced Computing and Communication Technologies, ICACCT2007, Panipat, Haryana. Pp.591-594.

[9] Mohd. Sadiq, and Shabbir Ahmed, "Estimation of the size of the software projects with high Complexity using Function point", 2nd National conference on Computing for Nation Development.

[10] Mohd. Sadiq, Danish Raza Rizvi, Suman Aggarwal, " Weaknesses of Software risk Estimation Models", 2nd National Conference on Emerging Trends in Computer Science and Information technology, AFSET, Faridabad. Pp (146-150)

[11] Mohd. Sadiq, Shabina Ghafir, Mohd. Shahid, "An Approach for Eliciting Software Requirements and its Prioritization using Analytic Hierarchy Process", IEEE International Conference on Advances in Recent Technologies in Communication and Computing, 2009, ACEEE annual world congress on Engineering and Technology , Kerala, India.

[12] Mohd. Sadiq, Shabina Ghafir, Mohd. Shahid," A Framework to Prioritize the software Requirements using Quality Function Deployment", National Conference on Recent Development in Computing and its Application, 2009, organized by Jamia Hamdard, Delhi, India.

[13] Mohd.Sadiq, M. Sarosh Umar, Jawed Ahmad, Danish Raza Rizvi, and Mohd.Shahid, "Function Point Measurement: An Empirical Study" , Proceedings of National Conference on Information Technology: Present Practices and Challenges (NCIT2007), p.p. 317-321, organized by Asia Pacific Institute of Management, New Delhi-110025 (31/08/2007 to 01/09/2007)

[14] Mohd.Sadiq, Shabbir Ahmed, "Relationship between Lines of Code and Function Point and its Application in the Computation of Effort and Duration of a Software using Software Equation", International Conference on Emerging Technologies and Applications in Engineering, Technology and Sciences , ICATETS2008, Rajkot, Gujrat.

[15] Mohd. Sadiq, Danish Raza Rizvi, Mohd Shahid, Safdar Tanveer , "Function Point computation of a software written in C/C++", National Conference on Mathematical Modeling, Optimization and Their Applications,OPTIMA-2007, Jointly Organized by BVICAM and Society for Reliability Engineering, Quality and Operation Management

[16] Ray C. Williams, George J. Pandelios, and Sandra G. Behrens, "Software Risk Evaluation (SRE) Method description (Version-2.0), Technical report December-1999.

[17] Say-Wei Foo , Armugam Muruganatham, " Software Risk assessment Model", ICMIT 2000, IEEE, pp-536-544.

[18] Susan A Sherer, "The Three dimensions of Software Risk: technical, Organizational, , and Environmental,", IEEE-2005.

[19] Low G.C. and Jeffery D.R.1990, Function Point in the Estimation and Evaluation of the Software Process, IEEE Trans Software Engineering, Vol.16, no.1.

[20] International Function Point User Group (IFPUG), Function Point Counting Practices Manual, Release 4.0, IFPUG, Westerville, Ohio, April 1990.

[21] Zuse H. 1991., Software Metrics-Methods to Investigate and Evaluate Software Complexity Measures, Proc. Second Annual Oregon Workshop on Software Metrics, Portland.

[22] Ho-Leung, TSOI, 2005, "To Evaluate the Function Point Analysis: A Case Study" , International Journal of the Computer , the Internet and Management Vol.13#1, Pp 31-40.

[23] Kemerer C.F.1987, An Emprical Validation of Software Cost Estimation Models, Comm.ACM, Vol.30, no.5.

[24] C.J. Lokan, 1999. An Emprical Study of the Correlations between Function Point Elements, Proc of 6-th International Symposium on Software Metrics.

[25] Ayad Ali Keshlaf and Khairuddin Hashim, "A Model and Prototype Tool to Manage Software Risks, pp 297-305, IEEE 2000.

[26] Mohd. Sadiq, Sunil, Aslam Qureshi, Mohammad Asim, Mohd. Shahid, " esrcTool: A Tool to Estimate the Software Risk and Cost" , IEEE Second International Conference on Computer Modeling and Simulation", organized by the International Association of Computer Science and Information Technology, Sanya, China (Accepted for Publication)