

# A Hybrid Approach to Sizing Problem in Software Project Estimation

Manas Kumar Yogi<sup>1</sup> Vijayakranthi Chinthala<sup>2</sup>

<sup>1</sup>Sr.Assistant Professor <sup>2</sup>M.Tech, Computer Science and Engineering Department

<sup>1</sup>Ellenki Engineering College <sup>2</sup>Indur Institute of Engineering and Technology  
Siddipet, Andhra Pradesh, India

**Abstract**--- Software Project Estimation is the process of predicting the most realistic use of effort required to develop or maintain software based on incomplete, uncertain and/or noisy input. Effort estimates may be used as input to project plans, iteration plans, budgets and investment analyses, pricing processes and bidding rounds. There are many ways of categorizing estimation approaches like Expert estimation where the estimate is produced based on judgmental processes. The Formal estimation model where the usage of a formula derived from historical data. Combination-based estimation is based on a judgmental or mechanical combination of estimates from different sources. We present a hybrid technique to the sizing problem in Estimation which is very much critical to a project's success.

**Keywords:** Estimation, Fuzzy, Function Point, Sizing Problem, Planning

## I. INTRODUCTION

Estimation is the first major activity for effective project planning. Estimation tries to find how much monetary resources, effort, time it will consume to construct a specific software based system. It is done by software project managers by having meaningful solicitations from concerned stakeholders and software engineers. Keeping in view the data of software metrics gathered from previous projects. It is important because no company has infinite amount of resources to build a software based system. It begins with a scope description of the software product. Then this problem of software construction is decomposed into a set of smaller problems and each of these is decomposed into a set of smaller problems and each of these is estimated with experience and historical data as the guiding factors. Before concluding a final estimate due Weight age is given to problem complexity, risk. We cannot ensure until project completion that our estimation procedure is correct or not. However if we follow a systematic approach, establish a realistic schedule and adapt continually as project moves ahead, we will feel confident that we have given our best shot. Availability of comprehensive software metrics for past projects help in estimating software with greater assurance, schedules are made to prevent past difficulties, thus reducing the overall risk. In Incremental process model for software development it's always possible to revisit the estimate and whenever user makes requirement changes, revise estimation also. The impact estimation revision will be high on stakeholders as variability in software requirements causes instability in cost and schedule of the project. The most crucial part of software estimation is it must be done within the 4 estimation is it must be done within the 4

dimensions of software feasibility i.e., technology, finance, time, resources.

## II. PROBLEM OF SOFTWARE SIZING

Estimation will be successful is software size is known beforehand. "Size" means a quantifiable outcome of software project. Below table enumerates software estimation options at acceptable risk: There are various approaches to sizing problem.

Sl. No	Options of reliable Cost, Effort estimate	Remark
1	Delay estimation until late in project	Not practical as Cost estimates should be provided upfront
2	Estimates can be based on similar completed projects	Reasonably well if current project is similar to past efforts and other project equivalents are roughly equivalent.
3	Simple decomposition techniques to generate project effort, cost estimates	Viable approach
4	Empirical models for software cost, effort estimation	Viable approach

Table 1: Approaches to Software Sizing Problem

### A. Four Different Approaches to the Sizing Problem Are

- 1) *Fuzzy Logic Sizing*: To apply this approach, the Planner must identify the type of application, establish its magnitude on a qualitative scale and then refine the magnitude within the original range.
- 2) *Function Point Sizing*: The Planner develops estimates of the information domain characteristics.
- 3) *Standard Component Sizing*: Software is made of numerous standard components generic to a specific application area. Here, the project planner estimates the number of occurrences of each standard component and consequently uses historical project data to determine the delivered size per standard component.
- 4) *Change sizing*: This strategy is used when a project includes the use of existing software that must be modified in some way as part of a project. The Planner estimates the number and type (e.g. Reuse, adding code, changing code, and deleting code) of modifications that must be done.

### III. PROPOSED METHODOLOGY

In this paper, a hybrid approach is proposed to estimate size of the project.

Combination of the properties of Fuzzy logic sizing and FP Sizing approaches to develop a hybrid estimation approach. Fuzzy Logic (FL) starts with the concept of fuzzy set theory. It is a theory of classes with un-sharp boundaries, and considered as an extension of the classical set theory. The membership  $\mu$  of an element  $x$  of a classical set  $A$ , as subset of the universe  $X$ , is defined as follows:

$$\mu_A(x) = 1 \text{ if } x \in A \quad (3.1)$$

$$\mu_A(x) = 0 \text{ if } x \notin A \quad (3.2)$$

In our proposed model we use  $\mu$  to indicate the values from 0 to 5 as developing optimistic (high), most likely, pessimistic (low) factors.

Each of complexity weighting factors is estimated and the value adjustment factor is computed as shown further.

On a scale of 0 to 5, we consider,  $\min=0$ ,  $\max=5$ , so, the membership factor,  $\mu$  is defined as

$$\mu = 1 - 2(\max - \text{value})/10 \quad (3.3)$$

Hence we get the following:

Value	Membership Value( $\mu$ )
5	1
4	0.8
3	0.6
2	0.4
1	0.2
0	0

Table 1: Correspondence between Weighting Factor and Fuzzy Values

SL.NO.	FACTOR	$\mu$
1	Backup and recovery	
2	Data communications	
3	Distributed Processing	
4	Performance Critical	
5	Existing Operating Environment	
6	Online data entry	
7	Input transaction over multiple screens	
8	Internal logic files updated online	
9	Information domain values complex	
10	Internal processing complex	
11	Code designed for reuse	
12	conversion/installation in design	
13	Multiple installations	
14	Application designed for Change	
	Value Adjustment Factor(VAF)	Total

Table 3: Factors for Calculating Function Points

Finally, we calculate the Expected Estimation Value, EEV as shown below:

$$EEV = \text{Count total } X [0.65 + 0.1 X (\sum \text{VAF})] \quad (3.4)$$

The Count total is computed as Sum of Number of External inputs and Number of external outputs and number of internal logic files and Number of external Interface files. The EVE gives an indication of organizational productivity within allocated resources.

### IV. FUTURE WORK

The proposed model can be applied to organizations which are developing e-commerce applications, real time applications or PC applications. The applicability of the model can be established by experimental results which will show the indispensable relationship between the value adjustment factors and software size under production. The complexity of the software and the software development methodology adopted will also impact the EEV. Nevertheless the proposed model will provide a research scope in this field.

### V. CONCLUSION

The proposed approach can be helpful for a software project planner to estimate the software size and consequently gives an empirical model based on which project risks can be reduced significantly. The inclusion of fuzzy logic in assigning the value adjustment factors can be very much helpful as it undertakes uncertainty measure while predicting the final value. With regard to the Function Point metric it makes our Expected Estimate Value much more robust.

### REFERENCES

- [1] A. J. Albrecht, "ensuring application development productivity", SHAR/GUIDIB Application development Symposium.
- [2] A. S. Andreou and E. Papatheocharous, "Software Cost Estimation using fuzzy Decision Trees", in 23rd IEEE/ACM International Conference on Automated Software Engineering, (2008), pp. 371-374.
- [3] I. Attarzadeh and S. Hockow, "Improving the Accuracy of Software Cost Estimation Model Based on a New Fuzzy Logic Model", World Applied Sciences Journal, vol. 8, no. 2, (2010), pp. 177-184.
- [4] B. Boehm, "Software Engineering Economics", Englewood Cliffs, NJ Prentice-Hall, (1981).
- [5] A. Idri, A. Abran and L. Kjiri, "COCOMO cost model using Fuzzy Logic", 7th International Conference on Fuzzy Theory & Technology Atlantic, New Jersey, (2000).
- [6] A. Idri, A. Abran and T. M. Khoshgoftaar, "Fuzzy Analogy: A new Approach for Software Cost Estimation", 11 International Workshop in Software Measurements, Montreal, (2001) August 28-29, pp. 93-101.
- [7] P.G. Bishop, R. Bloomfield, "A conservative theory for long term reliability growth prediction", IEEE Transactions on Reliability, Vol. 45, No.4, pp. 550-560, Dec. 1996.
- [8] S. Gokhale, W.E. Hong, K. Trivedi, J.R. Horgan, "An analytical approach to architecture based software reliability prediction", Proceedings of the 3rd IEEE International Computer Performance and Dependability Symposium, (IPDS-98), pp.13-22, 1998.