Estimation of Software Testing Effort: An Intelligent Approach

Praveen Ranjan Srivastava
Computer Science and Information System Group
Birla Institute of Technology and Science, Pilani, Rajasthan, India
praveenrsrivastava@gmail.com

ABSTRACT

Software Testing is an important process of software development that is performed to support and enhance reliability and quality of the software. Studies indicate that 40-50 percent of the cost of software development is devoted to testing, with the percentage for testing critical software being even higher. This poster makes an attempt to estimate reliable software testing effort using fuzzy logic.

Key words: Software Engineering, Software Development Effort, Software Testing Effort (STE), Line of Code (LOC).Confidence(C), Fuzzy Logic, Defuzzification, Test Effort Drivers (TEDs)

1. INTRODUCTION

Software testing is a necessary and important activity of software development process. Myers (1979) states that "Software Testing is the process of executing a program with the intent of finding errors". The importance of testing can be understood by the fact that "around 35% of the elapsed time and over 50% of the total cost are expending in testing programs. Software Development Effort (SDE) 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 several models for estimating software development effort. Some of them are COnstructive COst MOdel (COCOMO), Function point analysis and so on. The COCOMO is an algorithmic software cost estimation model developed by Barry Boehm in 1981. This model uses a basic regression formula, with parameters that are derived from historical project data and current project characteristics. COCOMO model is the most widely used model for estimating the software development effort .Once we have estimated the SDE, Software Testing Effort (STE) is estimated as 40 to 50 percent of the development .This poster deals with the features of the STE estimation problem by proposing a novel Fuzzy model by integrating COCOMO, fuzzy logic and weighing techniques (to account for the past experience and knowledge base which influence to software testing process), Test Effort Drivers (TEDs) into one platform. In this model triangular membership functions with monotonic constraints have been chosen, which finally achieve good generalization. The proposed model is finally validated with a case study of an Organization. Another feature of the present model is that it allows for continuous rating values and therefore avoids the problem of similar projects having large variances in cost estimations. Hence, in this poster try to estimate approximate Software Testing Effort (STE) value using fuzzy logic and proposed software Test Effort Drivers (TEDs).

2. PROPOSED APPROACH FOR ESTIMATING SOFTWARE TESTING EFFORT (STE)

An algorithm is constructed as given below in order to assess STE along with block diagram as shown in figure1

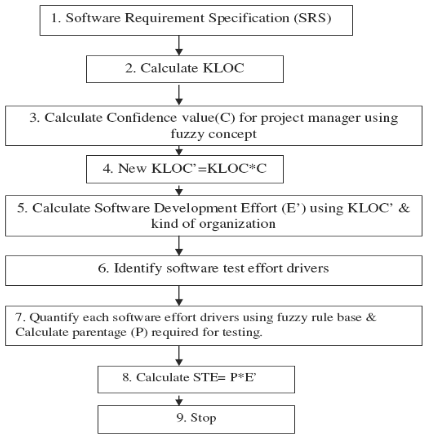

1. Software Requirement Specification (SRS)
2. Calculate KLOC
3. Calculate Confidence value(C) for project manager using fuzzy concept
4. New KLOC'=KLOC*C
5. Calculate Software Development Effort (E') using KLOC' & kind of organization
6. Identify software test effort drivers
7. Quantify each software effort drivers using fuzzy rule base & Calculate parentage (P) required for testing.
8. Calculate STE= P*E'
9. Stop

Figure1: [Flow chart of Calcuation of Software Testing Effort.]

3. APPROXIMATE ESTIMATION OF SOFTWARE DEVELOPMENT EFFORT (SDE)

Estimation of SDE with the help of COCMO is depending upon LOC which is generally estimated by senior project managers on the basis of his/her ability. By using fuzzy rule base, we find the exact confidence (C) value. Once you identified confidence then calculate exact size of code.

KLOC'=C*KLOC

Once a new project estimation size (KLOC') with full confidence is estimated, intermediate COCOMO model can be applied to calculate the approximate Software Development Effort (say E') which is given by

$E'=ai\ (KLOC')^{(bi)}.EAF$

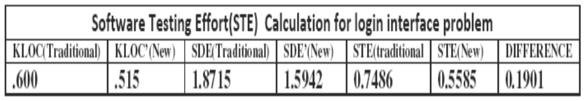
After evaluating Software Development Effort, the percentage of the software development effort supposed to be invested in testing should be determined. This percentage is purely based on heuristics and there is no standard procedure for arriving at this percentage as the process of software testing is a complicated task. However, it may vary between 40 to 50 percent as discussed in the previous section. Hence in the subsequent section, an attempt has been made to estimate the true value of percentage using fuzzy logic.

4. FACTORS ON WHICH TESTING EFFORT DEPENDS ON

Testing effort depends on several factors which are commonly known as Test Effort Drivers. These test effort drivers make software testing effort estimation accurately. In this study four prime factors have been identified as described in the tabular form. However, they may vary depending on type of problem.

Test Effort Drivers(TEDs)	Properties
Software Complexity(SC)	SC is directly proportional to STE.
Software Quality(SQ)	SQ is directly proportional to STE.
Schedule Pressure(SP)	SP is inversely proportional to STE.
Work Force Drivers(WFD)	WFD are directly proportional to STE.

After identifying TEDs, there is need to quantify it to get exact value for the estimation of STE. For the quantification of exact value of TEDs, again we have to use fuzzy rule, and we find out exact percentage value (P) of those test effort drivers.
We have tested our proposed method with one of real software and we find out the following outcomes.



Software Testing Effort(STE) Calculation for login interface problem						
KLOC(Traditional)	KLOC'(New)	SDE(Traditional)	SDE'(New)	STE(traditional	STE(New)	DIFFERENCE
.600	.515	1.8715	1.5942	0.7486	0.5585	0.1901

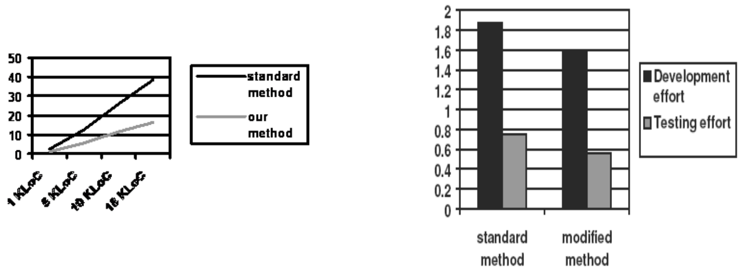


Figure 2: [Analysis of result]

We have conducted proposed method over several modules and several projects and we find out proposed method is much more valuable rather than traditional method, and we arrived at the following graph which shows the testing effort using the standard method and our method which incorporated various parameters.
We can see from Figure2 (GRAPH) as the number of lines of code is increasing, the difference between the standard method and our method is increasing. Hence proving proposed method to be more effective in places where the number of lines of code is very large.

5. CONCLUSIONS AND FUTURE SCOPE

This poster proposes a fuzzy based model for estimating approximate software testing effort. As fuzzy logic is powerful tool in solving real world problems with imprecise and uncertain information and in dealing with semantic knowledge, the presented model is easily capable of incorporating uncertainty due to the difficulty in exact quantification of certain efforts. The results obtained due to this process are very encouraging. However, one of the greatest difficulties in using the model is determining and fine-tuning of fuzzy rules which depends on the exposure and experience of the decision maker. In this model, fixed triangular membership functions have been considered for the analysis and fuzzy rules have been derived. These fuzzy rules express the information for interpretation of the nature of software testing efforts. The interpretation of each fuzzy rule is made by analyzing its basis and its output which finally provides a generalization capability within a domain. Finally, the introduction of fuzzy logic allows the integration of numerical data and expert knowledge and can be a powerful tool when tackling important problems in software engineering such as cost and quality prediction. In this study there are only four test effort drivers have been considered to analyze the software testing efforts. However, few more number of drivers can be added to make the problem even more realistic. Identification of factors is done by a team of experts, which is flexible depending on the nature of the software requirement. By this process software testing efforts can be evaluated more accurately which may ultimately help software developers to a greater extent. A number of extensions and applications of the model may be possible by using techniques like artificial neural networks, evolutionary computation and combination of neurons-fuzzy approach. These techniques can be used to model more complex nonlinear problems and in fact there is considerable need for applied research and strategy evaluation in this area using these techniques. This proposed model discussed here is further extended for computing pre and post STE for any organization.

REFRENCE:

- Pressman (2005), software engineering book, TMH.
- Zadeh, Lotfi A (1965), Fuzzy Sets, Information and Control 8, 338-353.
- Agrawal Manish and Chari Kaushal (2007), Software Effort, Quality, and Cycle Time: A Study of CMM Level 5 Projects. IEEE Transactions on Software Engineering, 33(3), 145-156.
- Praveen Ranjan Srivastava et al." Optimization of Software Testing Effort using Fuzzy Logic", International journal of Computer Sciences and Engineering Systems. (Accepted and to appear Vol. 3, No. 3 (July), 2009(in press)