IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

Fuzzy Logic in Software Effort Estimation: An Exploration

Archana Srivastava

Assistant Professor, Amity University, Lucknow

Abstract: The inaccuracy of software cost estimates has for long been a source of frustration to software practitioners and cost estimation researchers. Despite huge efforts to improve this important practice, estimation accuracy is still low. The accuracy of project estimates can have a dramatic impact on profitability. Software development projects are characterized by regularly over running their budgets and rarely meeting deadlines. Effective software estimation is one of the most important software development activities however it is also one of the most difficult tasks to estimate the accurate cost. Estimating is a critical business process, especially at the early stages of the project. This paper discusses & reviews the importance and need for accurate software project estimation and the problems associated with estimation. It also discusses that how uncertainty in software cost estimation can be reduced by using fuzzy logic approach.

Keywords: cost estimations, effort estimation, fuzzy.

1. INTRODUCTION

Software efforts estimation is one of important activity of software development. Software cost estimation plays an important role in software engineering practice, often determining the success or failure of contract negotiation and project execution. Cost estimation's deliverables such as effort, schedule, and staff requirements are valuable information for project formation and execution.

They are used as key inputs for:

- ❖ Project bidding and proposal
- ❖ Budget and staff allocation
- ❖ Project planning, progress monitoring and control
- ❖ Investment decision
- ❖ Tradeoff and risk analysis
- ❖ Stakeholder negotiation and expectations management

When we complete efforts estimation, we got the person - month or Person- hours required to build that project. Now we have to make a plan that will shows that which activity will be complete on what time and how much effort will be required to complete it.

2. REASONS FOR INACCURACY IN ESTIMATION

2.1 PROBLEMS WITH REQUIREMENTS:

Almost all organizations blame problems with requirements as major reasons for inaccurate cost estimates. The problems sited include; incomplete, incorrect, ambiguous, inconsistent and incomprehensible requirements. Some of the reasons for problems with requirements included the following:

- ❖ Users do not understand their requirements during early stages of the project.
- ❖ Correct and complete requirements for complex systems are impossible to achieve, especially at the early stages.
- ❖ Long development time, leading to requirements that are obsolete before the system is delivered.
- ❖ Large staff turnover for end users, resulting in changing requirements as new people arrive.

2.2 THE SOFTWARE PROCESS AND PROCESS MATURITY:

The software process and process maturity of an organization also affects its cost estimation processes. An organization cannot achieve accurate estimates if it has no clear idea of how it goes about its operational processes. Every development team with good management has developed an efficient software process, even though it might not be formalized, written down, and monitored, this process will be followed by the team. The state and maturity of any organizations software process, is also a major factor that affects it estimation process.

2.3 MONITORING PROGRESS OF PROJECT:

There is a need for all software development organizations to monitor the development process and measure progress made in order to be able to control software development costs.

2.4 LACK OF HISTORIC DATA:

Every organization needs information about previous projects in order to carry out accurate estimations for future development projects.
In the case of small organizations working with relatively constant work force and smaller projects, this may be achieved by relying on the knowledge and experience of key people in the organization.

For larger organizations, where the projects are multifaceted and knowledge is distributed among a larger group of people, relying on people's memory is not sufficient. In order to solve this problem of lack of

historical data, organizations can maintain a historical database of previous software cost estimates, and previous project performance evaluations. The accuracy of project estimates can have a dramatic impact on profitability. Software development projects are characterized by regularly over running their budgets and rarely meeting deadlines.

3. EFFECTIVE SOFTWARE ESTIMATION

Effective software estimation is one of the most important software development activities however it is also one of the most difficult. Under estimating a project will lead to **under staffing** it, **under scoping the quality assurance effort** and setting **too short a schedule**. That in turn can lead to **staff burnout, low quality**, **loss of credibility**, **deadlines being missed** and ultimately to inefficient development effort that takes longer than normal. Overestimating a project can be almost as bad. Parkinson's Law is that work expands until available time comes into play. This means that the project will take as long as estimated even if the project is over estimated. An accurate estimate is a critical part of the foundation of efficient software development.

Software effort estimation is one of the most critical and complex, but an inevitable activity in the software development processes. Over the last three decades, a growing trend has been observed in using variety of software effort estimation models in diversified software development processes. Along with this tremendous growth, it is also realized the essentiality of all these models in estimating the software development costs and preparing the schedules more quickly and easily in the anticipated environments.

Although a great amount of research time, and money have been devoted to improving accuracy of the various estimation models, due to the inherent uncertainty in software development projects as like complex and dynamic interaction factors, intrinsic software complexity, pressure on standardization and lack of software data, it is unrealistic to expect very accurate effort estimation of software development processes [1].

4. FUZZY LOGIC

In 1965, Lofti Zadeh formally developed multi-value set theory, and introduced the term fuzzy into the technical literature . 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 fuzzy logic model uses the fuzzy logic concepts introduced by Lotfi Zadeh [2]. Fuzzy reasoning consists of three main components: fuzzification process, inference from fuzzy rules and defuzzification process.

Fuzzification process is where the objective term is transformed into a fuzzy concept. The membership functions are applied to the actual values of variables to determine the confidence factor or membership function (MF). Fuzzification allows the input and output to be expressed in linguistic terms.

Inferencing involves defuzzification of the conditions of the rules and propagation of the confidence factors of the conditions to the conclusion of the rules. A number of rules will be fired and the inference engine assigned the particular outcome with the maximum membership value from all the fired rules.

Defuzzification process refers to the translation of fuzzy output into objective.

The membership $\mu_A(x)$ of an element x of a classical set A, as subset of the universe X, is defined as follows:

$\mu_A(x) = 1$ if $x \in A$

0 if $x \in A$

A system based on FL has a direct relationship with fuzzy concepts (such as fuzzy sets, linguistic variables, etc.) and fuzzy logic.

The popular fuzzy logic systems can be categorised into three types: Pure fuzzy logic systems, Takagi and Sugeno's fuzzy system, and fuzzy logic system with fuzzifier and defuzzifier [3].

Since most of the engineering applications produce crisp data as input and expects crisp data as output, the last type is the most widely used type of fuzzy logic systems. Fuzzy logic system with fuzzifier and defuzzifier, first, proposed by Mamdani and it has been successfully applied to a variety of industrial processes and consumer products [27]. The main three steps of applying fuzzy logic to a model are:

Step 1: Fuzzification: It converts a crisp input to a fuzzy set

Step 2: Fuzzy Rule-Based System: **Fuzzy logic systems** use fuzzy IF-THEN rules.

Fuzzy Inference Engine: Once all crisp input values are fuzzified into their respective linguistic values, the inference engine accesses the fuzzy rule base to derive linguistic values for the intermediate and the output linguistic variables

Step 3: Defuzzification: It converts fuzzy output into crisp output

Most of the software estimates should be performed at the beginning of the life cycle, when we do not yet know the problem we are going to solve. Effort estimation is used to predict how many hours of work and how many workers are needed to develop a project.

Fuzzy systems try to emulate cognitive processes of the brain with a rule base. The basic concept is inspired by the human processes where the decisional criteria are not clear cut, but blurred and it is difficult to find objective to make the decisions more precise and clear. Fuzzy decision systems are based on fuzzy logic that tries to reproduce the fuzzy human reasoning.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 5, May 2016

5. FUZZY LOGIC IN SOFTWARE EFFORT ESTIMATION

Since many of the independent variables in software metric models are either difficult to quantify (for example complexity), or are only known to a rough degree (such as system size), the use of fuzzy variables seems intuitively appealing. It is our conjecture here that project managers are in fact able to classify systems using fuzzy variables with reasonable levels of both accuracy and consistency. While complexity can be defined in an algebraic sense, and in fact it has been defined in a large number of ways in the past, such formal definitions are always to some extent arbitrary. It is instead suggested here that complexity is a multifaceted concept, but that experienced project managers would be able to make fairly consistent classifications of projects in terms of one or a small number of types of complexity. This has the added advantage of reducing the number of variables used as inputs into the model.

6. REDUCING COMMITMENT THROUGH FUZZY OUTPUTS

Particularly at the very early stages of a software development project, estimating to within one person-hour or person-day is simply not realistic. Instead, a fuzzy system may be used to transform linguistic labels, or numerical values, indicating system size and complexity, personnel experience, and other factors of influence into an equally imprecise label indicating predicted effort, for example very high. While this approach may well be imprecise, this is justified and should ensure that personnel associated with the project do not attach unwarranted accuracy to the figures produced. As the project progresses and a greater degree of certainty is established in relation to the scope of the project (and also as data starts to become available), then more precise indicators of effort may be formulated, either through more and smaller membership functions or allowing for numerical defuzzification.

The most significant activity in software project management is Software development effort prediction. The literature shows several algorithmic cost estimation models such as Boehm's COCOMO, Albrecht's' Function Point Analysis, Putnam's SLIM, ESTIMACS etc., but each model do have their own pros and cons in estimating development cost and effort. This is because project data, available in the initial stages of project is often incomplete, inconsistent, uncertain and unclear. The need for accurate effort prediction in software project management is an ongoing challenge.

A fuzzy model is more apt when the systems are not suitable for analysis by conventional approach or when the available data is uncertain, inaccurate or vague. Fuzzy logic is a convenient way to map an input space to an output space. Fuzzy Logic is based on fuzzy set theory.

A fuzzy set is a set without a crisp, clearly defined boundary. It is characterized by a membership function, which associates with each point in the fuzzy set a real number in the interval [0, 1], called degree or grade of membership. The membership functions may be Triangular, GBell, Gauss and Trapezoidal etc.

7. LITERATURE REVIEW OF FUZZY LOGIC IN SOFTWARE EFFORT ESTIMATION

[4] Intelligent Systems provide alternative paradigms aimed at facilitating the representation and manipulation of uncertain, incomplete, imprecise or noisy data. Specifically, Fuzzy Logic offers a particularly convenient way to generate a keen mapping between input and output spaces because of fuzzy rules' natural expression [5]. Fuzzy logic with its offerings of a powerful linguistic representation can represent imprecision in inputs and outputs, while providing a more expert knowledge based approach to model building.

A study by Hodgkinson and Garratt claims that estimation by expert judgment was better than all regression based models [6]. One of the major researches into fuzzy logic application to cost estimation is that of MacDonell et al.[7]. Their approach, starting from [8], took a total leap from application of fuzzy logic to already existing regression-based model, but an expert knowledge based application of fuzzy logic. This particular research has evolved into the development of a tool, FULSOME, to assist project managers in estimation.

FULSOME was developed [9, 10, 11] using fuzzy logic to help software metricians in data acquisition, model expression and knowledge gathering issues. It provides interfaces for data entry, membership function construction, rule creation and output of the inference process, in similar fashion to the MATLAB Fuzzy Logic toolbox [12]. The experts can perform the membership function definition and rules manually, as it is the case in the MATLAB toolbox. The rules generated are also not sensitive to users needs. When user supplies rules in a bid to extend the initial rules generated, it will destroy their clustering method because the added rules were not used in the initial clustering to extract membership functions. Using the system requires historical data or the availability of experts to supply the data based on their experience. It allows software managers to have the flexibility to identify and define features believed to affect software projects.

Attempts have been made to fuzzify some of the existing algorithmic models in order to handle uncertainties and imprecision problems surrounding such models. The first realization of the fuzziness of several aspects of one of the best known [13], most successful and widely used model for cost estimation, COCOMO, was that of Fei and Liu [14], where they introduced fuzzy set theory in their work on f-COCOMO. They observed that an accurate estimate of delivered source instruction (KDSI) cannot be done before starting a project, and it is unreasonable to assign a determinate number for it. Jack Ryder [38] investigated the application of fuzzy modeling techniques to two of the most widely used models for effort prediction; COCOMO and the Function-Points models respectively.

DOI 10.17148/IJARCCE.2016.55111

Musflek et al.[15] fuzzify the basic COCOMO model at two different levels of detail. The first level called f-COCOMO is concerned about representing size of software project as fuzzy set while the coefficients representing mode remain crisp values. The second level called the augmented f-COCOMO provides a representation of the modes of software development as singleton fuzzy sets. The proposed work is an algorithmic cost model; the basic COCOMO model is used as underlying model. The input required is size measured in KLOC and mode. The size representation for two levels are based on the fuzzy sets, the approach is able to reduce the sensitivity to imprecision in the input data although, the discrete number representation of mode does not take care of imprecision. It covers a wide range of systems because it takes mode of software development into consideration. The approach is not extendible since it is strictly dependent on the existence of the underlying COCOMO model and the input to the estimation system must be LOC and mode.

Idri and Abran [16] applied fuzzy logic to the cost drivers of intermediate COCOMO model. They presented a two stage implementation called simple F-COCOMO model and augmented F-COCOMO model respectively. Idri and Abran [40] fuzzified the cost drivers of intermediate COCOMO 81's model to take care of the very sharp transition between two different intervals defined for a single cost driver. Fuzzy sets were defined to model the different categories, such that two analogous projects, in different categories, would not have a very large difference in their effort estimates.

The approach was more of a sensitivity analysis on the cost drivers, so fuzzyfication is only applied to the cost drivers and not to the other inputs. The application of fuzzy logic to represent the mode and size as input to COCOMO model was later presented in [15].

Ahmed, M.A., Omolade Sailu et. al.[17] presented an adaptive fuzzy logic framework for software effort prediction. Their experiment was carried out on artificial datasets as well as COCOMO'81 public dataset. They reported promising experimental results despite of little background knowledge in the rulebase and training data. It does signify that there are potentials for improvements when the framework is deployed in practice, since experienced experts could augment with their knowledge.

Da yang et al.[18] proposed an extension to COCOMO II named COCOMO-U, in which Bayesian Belief Network (BBN) is used to extend the COCOMO II for cost estimation with uncertainty. COCOMO-U takes the probability distributions of the estimated project size and other 22 cost factors as input and produces the probability distribution of software development effort contrary to the most likely effort as in COCOMOII.

Marcio Rodrige Braz et al.[19] applied the concept of fuzzy set theory to develop a metric Fuzzy Use Case Size Point (FUSP) based on Use Case Size Points (USP) for the effort estimation of object-oriented software. They used the concept of Fuzzy Function Point Analysis (FFPA) in their extended model named FUSP. The proposed model was validated through a project database of a private company. Although FUSP presented better result than USP but for certain modules had worst results, due to the underlying uncertainty at the early phases of a project. Fuzzy Logic has also offered itself as a useful tool to aid other techniques for software cost estimation like analogy. Similarity between projects is often used when estimating software effort by analogy.

Various authors have put forward various proposals for means of deriving similarity as input to the estimation process; the nearest neighbor algorithm [20] is one such approach. This algorithm cannot handle projects attributes described by categorical variables other than binary valued variables. An alternative approach using fuzzy logic was proposed by Idri et al.[20,21] to deal with this limitation. Evolutionary computation has also recently found its usefulness in software effort estimation. Burgess et al.[22] applied genetic programming (GP), an application of GA, to software effort estimation. Based on reasoning by analogy, fuzzy logic and linguistic quantifiers, Idri et al.[26] proposed another approach to estimate the software project effort. Their approach can be used when software projects are described by categorical and/or numerical data, thus improved the classical analogy approach which does not consider categorical data.

The advantages of proposed approach are twofold that is to handle the imprecision and uncertainty when describing software projects, also by using Regular Increasing Monotone (RIM) linguistic quantifier to guide the aggregation of the individual similarities between two projects. The author claimed that Fuzzy Analogy approach can be easily adapted and configured according to the needs of the target environment, further it can learn from previous situations.

Bayesian analysis, now considered as part of the constituents of soft computing, was used by Chulani et al.[23] to calibrate the 1998 version of the COCOMO II model to 161 data points. On comparing with the 1997 calibration using multiple regressions, the Bayesian approach was adjudged to perform better and more robust. Bayesian analysis was also used in the calibration of the 2000 version of COCOMO II [23], resulting in a higher predictive accuracy as well.

Harish Mittal and Pradeep Bhatia [27] presented two models for optimization of effort for specific application, based on fuzzy logic sizing; rather than using a single number they took software size (KLOC) as a triangular number. Empirical study was done not only on the 10 projects of NASA but also compared their results to the existing models

(bailey basili model, Alaa F. Sheta G.E. Model, and Alaa F. Sheta Model2), comparative study showed better results to some earlier models. They also claimed that the

methodology proposed is general enough to be applied to other models based on function point methods and to other areas of quantitative software engineering.

8. CONCLUSION

On the basis of this study we can say that selecting correct estimation is very difficult task. If we do any minor mistake for this work, result is high financial loss and completion of project time is increased. It is very important to continually re-estimate cost and to compare targets against actual expenditure at each major milestone. This keeps the status of the project visible and helps to identify necessary corrections to budget and schedule as soon as they occur.

Fuzzy estimation method has its own strengths and weaknesses. A key factor in selecting a cost estimation model is the accuracy of its estimates. Unfortunately, despite the large body of experience with estimation models, the accuracy of these models is not satisfactory. Still lots of research are required in this area to generalize any method of estimation.

REFERENCES

[1] Satyananda, "An Improved Fuzzy Approach for COCOMO's Effort Estimation Using Gaussian Membership Function" Journal of Software, vol 4, pp 452-459, 2009

[2] L. A. Zadeh. "Fuzzy Sets". Information And Control, Vol. 8, 1965, pp. 338-353.

[3] A. Lotfi Zadeh, "The future of soft computing," The 9th IFSA World Congress and 20th NAFIPS International Conference, Vancouver, Canada, pp. 217-228, 2001.2464 [28] Conte, S., Dunsmore, H. and Shen, V.: "Software Engineering Metrics and Models", Benjamin-Cummings, Menlo Park, CA, 1986.

[4] Software Development Effort Estimation Using Fuzzy Logic - A Survey M. Wasif Nisar1,3 Yong-Ji WANG2 Manzoor Elahi3, 978-0-7695-3305-6/08, 2008 IEEE DOI 10.1109/FSKD.2008.370

[5] Hodgkinson, A.C. and Garratt, P.W.: "A NeuroFuzzy Cost Estimator" Proceedings of the 3rd International Conference on Software Engineering and Applications - SAE, pp. 401-406, 1999.

[6] MacDonell, S.G. Gray, A. R. and Calvert, M.J.: "FULSOME: A Fuzzy Logic Modeling Tool for Software Metricians" Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society - NAFIPS, IEEE, pp. 263-267, 1999.

[7] Gray, G. and MacDonell, S.: "Applications of Fuzzy Logic to Software Metric Models Development Effort Estimation" Proceedings of the 1997 Annual Meeting of the North American Fuzzy Information Processing Society - NAFIPS, Syracuse NY, USA, IEEE, pp. 394-399, 1997.

[8] MacDonell, S.G. and Gray, A.R.: "A Comparison of Modeling Techniques for Software Development Effort Prediction" Proceedings of the 1997 International Conf. On Neural Information Processing and Intelligent Information Systems, Dunedin, New Zealand, Springer-Verlag, 869-872, 1997.

[9] MacDonell, S.G. Gray, A. R. and Calvert, M.J.: "FULSOME: A Fuzzy Logic Modeling Tool for Software Metricians" Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society - NAFIPS, IEEE, pp. 263-267, 1999.

[10] [10] MacDonell, S.G. Gray, A. R. and Calvert, M.J.: "FULSOME: A Fuzzy Logic for Software Metric Practitionersand Researchers" Proceedings of the 6th International Conference on Neural Information Processing - ICONIP, IEEE, pp. 308-313, 1999.

[11] "Fuzzy Logic Toolbox User's Guide" MATLAB 6.0, Mathorks, Inc., 2000.

[12] Kirsopp, C. and Shepperd, M.J.: "Making Inferences with Small Numbers of Training Sets", 6th International Conference on Empirical Assessment and Evaluation in Software Engineering, Keele University, Staffordshire, UK, April 8th - 10th, 2002.

[13] Zonglian, F. and Xihui L.: "f-COCOMO: Fuzzy Constructive Cost Model in Software Engineering", Proc. of IEEE Int. Conf. On Fuzzy Systems, IEEE, pp. 331-337, 1992.

[14] Musilek, P. Pedrycz, W. Succi, G. and Reformat, M.:"Software Cost Estimation with Fuzzy Models" Applied Computing Review, Vol. 8, No. 2, pp. 24 - 29, 2000.

[15] Idri, A. and Abran, A.: "COCOMO Cost Model Using Fuzzy Logic" 7th International Conference on Fuzzy Theory and Technology, Atlantic City, New Jersey, March 2000.

[16] Moataz A. Ahmed, Moshood Omolade Saliu, Jarallah AlGhamdi: "Adaptive fuzzy logic-based framework for software development effort prediction" Information and Software Technology, ELSEVIER 47, pp. 31-48, 2005.

[17] Da Yang, YuxiangWan, Zinan Tang, ShujianWu, Mei He and Mingshu Li: "COCOMO-U: An Extension of COCOMO II for Cost Estimation with Uncertainty" Q. Wang, Dietmar Pfahl, David M.Raffo and PaulWernick (Eds.): Software Process Change, LNCS 3966, pp. 132-141, 2006.

[18] Marcio Rodrigo Braz and Silvia R. Vergilio: "Using Fuzzy Theory for Effort Estimation of Object-Oriented Software" Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2004.

[19] Idri, A. and Abran, A.: "A Fuzzy Logic Based Set of Measures for Software Project Similarity: Validation and Possible Improvements" Proceedings of METRICS 2001, London, England, 2001, pp. 85-96.

[20] Idri, A. and Abran, A.: "Evaluating Software Project Similarity by using Linguistic Quantifier Guided Aggregations" Proceedings of IFSA/NAFIPS 2001, Vancouver, Canada, pp. 470 - 475, 2001.

[21] Burgess, C. J. and Lefley M.: "Can Genetic Programming improve software effort estimation? A comparative evaluation" Information and Software Technology, ELSEVIER 43, pp. 863-873, 2001.

[22] Conte, S., Dunsmore, H. and Shen, V.: "Software Engineering Metrics and Models", Benjamin-Cummings, Menlo Park, CA, 1986.

[23] X. huang, J. ren and L.F. Capretz.: "A neuro-Fuzzy tool for Software Estimation" Proceedings of the 20th IEEE International Conference on Software Maintenance, pp.520, 2004

[24] Cuauhtemoc Lopez Martin, Jerome Leboeuf pasquier, Cornelio Yanez M. and Agustin Gutierrez T.: "Software

[25] Development Effort Estimation Using Fuzzy Logic: A Case Study" Proceedings of the 6th IEEE Mexican International Conference on Computer Science, pp.113-120, 2005.

[26] Idri, A., Abran, A. and Taghi M. Khosgoftaar: "Fuzzy Analogy: A New Approach for Software Cost Estimation" International Workshop on Software Measurement, Montreal, Quebec, Canada, August 2001.

[27] Harish Mittal, Pradeep Bhatia: "Optimization Criterion for Effort Estimation using Fuzzy Technique" CLEI Electronic Journal, Vol. 10 Num. 1 Pap. 2, 2007.