International Journal of Technical Research and Applications e-ISSN: 2320-8163, www.ijtra.com,
Volume 5, Issue 2 (March - April 2017), PP. 68-72

"DEVELOPMENT OF A FRAMEWORK FOR SOFTWARE COST ESTIMATION: DESIGN PHASE"

[1]Syed Ali Mehdi Zaidi, [2]Dr. Vinodani Katiyar, [3]Prof. (Dr.) S. Qamar Abbas,
[1]Research Scholar, S. V. University, Gajraula, Amroha,
[2]Professor, SRM, University, Lucknow,
[3]Director, AIMT, Lucknow,
Uttar Pradesh, India

***Abstract*— Today's Software development cost estimation models are based on soft computing techniques as neural network, genetic algorithm, the fuzzy logic modelling etc. for finding the accurate predictive software development effort and cost estimation. Genetic Algorithm can offer some significant improvements in accuracy and has the potential to be a valid additional tool for software effort estimation. Genetic Algorithm is one of the evolutionary methods for the effort estimation.**

This paper focuses on the development of a framework for software cost estimation model in the design phase and variations of this model due to the public description of the algorithm, available data, as well as prior use and research by the research client by using object oriented software metrics. The framework will then be implemented for the development of a prediction model using Genetic Algorithm optimization techniques.

Till date many developments have introduced various methods of cost estimation including the recent advances in the field. These were based on the effort, size, schedule and other. But here effort will be made to evolve a framework for cost estimation keeping in mind object oriented perspective. There are many sophisticated parametric framework for estimating the size, cost, and schedule of Object Oriented software project.

***Index Terms* — a framework for software, estimation models.**

I. INTRODUCTION

A. Cost Estimation

Software cost estimation is a complex activity that requires knowledge of a number of key attributes about the project for which the estimate is being constructed. Cost estimating is sometimes termed "parametric estimating" because accuracy demands understanding the relationships among scores of discrete parameters that can affect the outcomes of software projects, both individually and in concert. Creating accurate software cost estimates requires knowledge of the following parameters:

1. The sizes of major deliverables, such as specifications, source code, and manuals.
2. The rate at which requirements are likely to change during development.
3. The probable number of bugs or defects that are likely to be encountered.
4. The capabilities of the development team.
5. The salaries and overhead costs associated with the development team.
6. The formal methodologies that are going to be utilized (such as the Agile methods).
7. The tools that are going to be utilized on the project
8. The set of development activities that are going to be carried out.
9. The cost and schedule constraints set by clients of the project being estimated.

Although the factors that influence the outcomes of software projects are numerous and some are complex, modern commercial software cost-estimation tools can ease the burden of project managers by providing default values for all of the key parameters, using industry values derived from the integral knowledge base supplied with the estimation tools. The several approaches for the cost estimation techniques are developed. It is classified into following: Model Based SLIM, checkpoint, SEER, COCOMO, Expertise Based-Delphi, and Rule-Based, Dynamics-Based, Abdel-Hamid Madnick.

Learning Oriented Neural, Case based, Regression Based, OLS, Robust, Composite, Bayesian, and COCOCMO-II. Each technique has their own significance and even its disadvantages are also highlighted. This paper concludes that no one model or single method should be favoured over others. The key to achieve the goal i.e. estimation, can be done through variety of tools and methods and then work upon the area that what reasons effects estimation.

B. Object-Oriented Technology (OOT)

Object-oriented technology, aims to overcome most of the problems associated with the traditional software technologies.

Reusability, high modularity, and the innovative approach to design, are expected to increase productivity in the production process. However, the criticality of cost estimation is increased by the change in the technological paradigm. Moreover, the existing techniques were developed according to the traditional software process and languages. The rapid growth of the object-oriented industry and the big capitals committed by many companies' calls for innovative models.

Object oriented design restrict the design space exclude procedure-oriented and other non-object software structures. Freedom to experiment with non-object-oriented designs is specifies to guide the designer to words an object – oriented solution. There is strong evidence that object - oriented models are generally natural and effective. However, object-oriented designs is a heuristic modelling technique that have proved conceptually power full practically effective, but may in some cases be suboptimal.

C. Genetic Algorithms (GA)

Genetic Algorithms (GA) are direct, parallel, stochastic method for global search and optimization, which imitates the evolution of the living beings, described by Charles Darwin. GA is part of the group of Evolutionary Algorithms (EA). The evolutionary algorithms use the three main principles of the natural evolution: reproduction, natural selection and diversity of the species, maintained by the differences of each generation with the previous. Genetic Algorithms works with a set of individuals, representing possible solutions of the task. The selection principle is applied by using a criterion, giving an evaluation for the individual with respect to the desired solution. The best-suited individuals create the next generation. The large variety of problems in the engineering sphere, as well as in other fields, requires the usage of algorithms from different type, with different characteristics and settings.

Genetic algorithms are a type of optimization algorithm, meaning they are used to find the optimal solution to a given computational problem that maximizes or minimizes a particular function. Genetic algorithms represent one branch of the field of study called evolutionary computation, in that they imitate the biological processes of reproduction and natural selection to solve for the `fittest' solutions [1]. Like in evolution, many of a genetic algorithm's processes are random, however this optimization technique allows one to set the level of randomization and the level of control [1]. These algorithms are far more powerful and efficient than random search and exhaustive search algorithms, yet require no extra information about the given problem. This feature allows them to find solutions to problems that other optimization methods cannot handle due to a lack of continuity, derivatives, linearity, or other features.

D. Research Methodology

The purpose of research is to discover answers to questions through the application of scientific procedures. The main aim of research is to find out the truth which is hidden and which has not been discovered as yet. Though each research study has its own specific purpose, we may think of research objectives as development of a cost estimation framework for software project. My aim is to develop effective approaches in the cost estimation framework for object oriented design phase.

Effective software project estimation is one of the most challenging and important activities in software development. Estimation is one of the cornerstones of effective project planning: effective project planning and control is not possible without a sound and reliable estimate.

The proposed work is to develop cost estimation framework in design phase using object oriented perspective as well as to develop a predictive model using a combination of Regression and Genetic Algorithm optimization techniques.

II. Related Research Works

A number of studies have been published to address cost estimation models and framework for software development and design phase. Existing studies are investigated and their contents and limitations are as follows:

- Pushpendra K Rajput, Geeta Sikka, and Aarti, (2014), proposed a hybrid model that exploits the uncertainty using clustering the data. In this proposed model they used Genetic Algorithm (GA) combined with COCOMO model on clustered data. Model carries the desirable features of neural network, including learning ability to classify the new project for using the COCOMO model with best fit parameters. The best parameters of COCOMO model can be found for each cluster. They made comparison of estimated effort with original COCOMO model which can be applied on larger data sets. This scheme also avoids the problem of different estimated cost of similar projects.
- Lalit V. Patil, et. Al., (2014) there are so many models available categorized into algorithmic and non-algorithmic model each of their strengths and weakness. The authors proposed a hybrid approach, which consists of Functional Link Artificial Neural Network (FLANN) and COCOMO-II with training algorithm. FLANN reduces the computational complexity in multilayer neural network. It does not have any hidden layer, and it has fast learning ability.
- K.Ramesh, et. al., (2013), analyzed algorithmic modes and non-algorithmic models in the existing models and provided in depth review of software and project estimation techniques existing in industry and literature based on the different test datasets along with their advantages and disadvantages.
- Rahul Chaudhary, et. al., (2013), showed that they can estimate and compare the cost and effort more accurately by using three technologies which are very prominent; they are Grouping Methodologies, Object Oriented Metrics and COCOMO II. All this work helps a manager or Estimator or User of Software to use the previous work (project) in new Real Time Project i.e. there are many references available for

continuing to new Real Time Project and secondly, when same goal project is developed by two different logics, then this Tool helps to compare between both Real Time Project in a single Dynamic window on the basic of Object Oriented Metric. They further compared the software project cost estimation methods based on grouping/groups as new methods that estimates software project cost accurately and are then compared between both the window (or projects result) result and help to fetch out more accurate and correct comparison that helps user/manager to select best old work for their future project.

• Tharwon Arnuphaptrairong, (2012), analyzed software sizing articles reviewed from the literature and presented the development, and achievements of software size measurement. From the literature review it was found that technologies and techniques related to requirement gathering, and software analysis and design, such as, Structured Analysis and Design Method (SSADM), and Object-oriented Analysis and Design (OOAD), had impacted on the size measurement models. This is because they are directly related to the software functionality. Significant future challenges for software sizing is probably the sizing for new product forms which include requirement or architectural specifications, stories and component-based development. They concluded that besides the new product forms, the new process forms.

• Gurdev Singh, et. al., (2011) studied different type of software metrics which are used during the software development. They showed that a metrics program that is based on the goals of an organization will help communicate, measure progress towards, and eventually attain those goals. People will work to accomplish what they believe to be important. Well-designed metrics with documented objectives can help an organization obtain the information it needs to continue to improve its software products, processes, and services while maintaining a focus on what is important. A practical, systematic, start-to-finish method of selecting, designing, and implementing software metrics is a valuable aid. • Mahmoud O. Elish and Karim O. Elish, (2009) used TreeNet in predicting object-oriented software maintainability. It has applied TreeNet using two datasets and compared its prediction performance against recently published object-oriented software maintainability prediction models (MARS, MLR, SVR, ANN, and RT). The results indicate that improved, or at least competitive, prediction accuracy has been achieved when applying the TreeNet model. The TreeNet model has achieved improved prediction accuracy in terms of Pred(0.25) and Pred(0.30) in both datasets. Furthermore, the TreeNet model has achieved the best MMRE in one dataset, and the second best MMRE in the other dataset. The results therefore reveal the effectiveness of TreeNet in predicting object-oriented software maintainability, and thus suggest that it can be a useful and practical addition to the framework of software quality prediction.

III. Applications of Genetic Algorithms

The algorithm described above is very simple, but variations on this basic theme have been used in a large number of scientific and engineering problems and models, including the following:

• Optimization: GAs has been used in a wide variety of optimization tasks, including numerical optimization as well as combinatorial optimization problems such as circuit layout and job-shop scheduling.

• Automatic Programming: GAs has been used to evolve computer programs for specific tasks, and to design other computational structures, such as cellular automata and sorting networks.

• Machine learning: GAs has been used for many machinelearning applications, including classification and prediction tasks such as the prediction of weather or protein structure. GAs have also been used to evolve aspects of particular machine-learning systems, such as weights for neural networks, rules for learning classifier systems or symbolic production systems, and sensors for robots.

• Economic models: GAs has been used to model processes of innovation, the development of bidding strategies, and the emergence of economic markets.

• Immune system models: GAs has been used to model various aspects of the natural immune system including somatic mutation during an individual's lifetime and the discovery of multi-gene families during evolutionary time.

• Ecological models: GAs has been used to model ecological phenomena such as biological arms races, host-parasite coevolution, symbiosis, and resource flow in ecologies.

• Population genetics models: GAs have been used to study questions in population genetics, such as Under what conditions will a gene for recombination be evolutionarily viable?".

• Interactions between evolution and learning: GAs have been used to study how individual learning and species evolution one another.

• Models of social systems: GAs has been used to study evolutionary aspects of social systems, such as the evolution of cooperation, the evolution of communication, and trailfollowing behaviour in ants.

• Cost Estimation: Software cost estimation is important for budgeting, risk analysis, project planning and software improvement analysis. There are numerous estimation techniques. During the past three decades there had been some significant developments in effort estimation, size of software and cost estimation methodology. Current software cost estimation models have been experiencing increasing difficulties in estimating the costs of software, as new software development methodologies and technologies are emerging very rapidly. Most of the software cost models generally rely on such inputs as estimates of lines of source code, delivered sets of instructions, function points and processing complexity or experience levels to produce cost estimates. These models

generally produce inaccurate results when used to estimate the cost of software development in current development environments such as those that use component-based software development environments like visual languages

IV. Proposed framework for prediction of software cost estimation

The present work deals with the development of a GA based optimization model for the prediction of the software cost estimates for which the object oriented dataset from forty Java systems derived during two successive semesters of graduate courses on Software Engineering is going to be used. For this a two stage data analysis will be done. Initially the dataset will be pre-processed, for the detection of the outliers using Robust Linear Regression Technique.

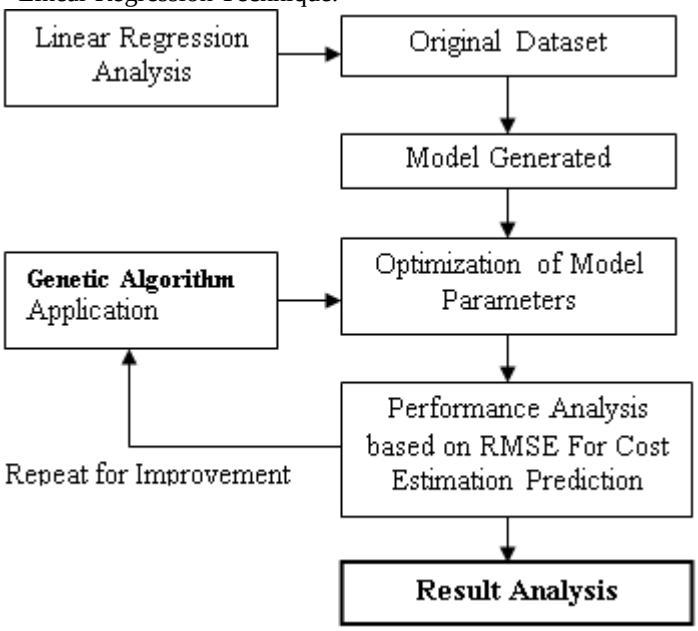

Figure 1: Proposed framework

This will be done by assigning a weight to each data point. Weighting is done automatically and iteratively using a process called iteratively reweighted least squares. In the first iteration, each point is assigned equal weight and model coefficients are estimated using ordinary least squares. At subsequent iterations, weights are recomputed so that points farther from model predictions in the previous iteration are given lower weight. Model coefficients are then recomputed using weighted least squares. The process continues until the values of the coefficient estimates converge within a specified tolerance.

Next, the model so obtained will be later on subjected to optimization of its model parameters using Genetic Algorithm optimization technique so as to arrive at a better software cost estimation prediction accuracy. The genetic operators such selection, crossover and mutation shall be used. GA runs to generate solutions for successive generations. Hence the quality of the solutions in successive generations improves. The process is terminated when an optimum solution is found. Finally, the performance of the model shall be analysed based on RMSE factor. The general framework for the present work is given above.

V. Algorithm for Software Cost Estimation Framework:

The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. You can apply the genetic algorithm to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, nondifferentiable, stochastic, or highly nonlinear. The genetic algorithm uses three main types of rules at each step to create the next generation from the current population:

- Selection rules select the individuals, called parents that contribute to the population at the next generation.
- Crossover rules combine two parents to form children for the next generation.
- Mutation rules apply random changes to individual parents to form children.

The following outline summarizes how the genetic algorithm works:

- The algorithm begins by creating a random initial population.
- The algorithm then creates a sequence of new populations. At each step, the algorithm uses the individuals in the current generation to create the next population.

To create the new population, the algorithm performs the following steps:

1. Scores each member of the current population by computing its fitness value.
2. Scales the raw fitness scores to convert them into a more usable range of values.
3. Selects members, called parents, based on their fitness.
4. Some of the individuals in the current population that have lower fitness are chosen as elite. These elite individuals are passed to the next population.
5. Produces children from the parents. Children are produced either by making random changes to a single parent mutation or by combining the vector entries of a pair of parents crossover.
6. Replaces the current population with the children to form the next generation.
7. The algorithm stops when one of the stopping criteria is met.

CONCLUSION

GA is a robust and general technique. Its main advantages over other local search methods are its flexibility and its ability to approach global optimality. The algorithm is quite versatile since it does not rely on any restrictive properties of the model. GA methods are easily "tuned". For any reasonably difficult nonlinear or stochastic system, a given optimisation algorithm can be tuned to enhance its performance and since it takes time and effort to become familiar with a given code, the ability to tune a given algorithm for use in more than one problem should be considered an important feature of an algorithm. Its convergence to the optima is 'good' even if the initial guess is far away from optima. It statistically guarantees finding an optimal solution.

In this study, applicability and capability of Genetic Algorithm techniques for application in software design cost estimation as a predictive tool has been investigated. It is seen that GA models are very robust, characterised by fast computation, capable of handling the noisy and approximate data that are typical of data used here for the present study.

From the analysis of the results given earlier it is seen that GA has been able to perform well for the prediction of effort estimation. Due to the presence of non-linearity in the data, it is an efficient quantitative tool. The studies have been carried out using MATLAB simulation environment.

REFERENCES

[1] Pushpendra K Rajput, Geeta Sikka, and Aarti, (2014), "CGANN-Clustered Genetic Algorithm with Neural Network for Software Cost Estimation", International Conference on Advances in Engineering and Technology (ICAET'2014), pp. 268-272.

[2] Mogili Umamaheswara Rao, et. al. (2014), "Effort Estimation for Object-Oriented System Using Artificial Intelligence Techniques", Volume No: 1(2014), Issue No: 10, pp. 248-252.

[3] N. Veeranjaneyulu, S.Suresh, Sk.Salamuddin3 and Hyejin Kim, (2014), " Software Cost Estimation on e-Learning Technique using A Classical Fuzzy Approach", International Journal of Software Engineering and Its Applications Vol. 8, No. 11 (2014), pp. 217-222.

[4] Lalit V. Patil, et. Al., (2014), "Develop Efficient Technique of Cost Estimation Model for Software Applications", International Journal of Computer Applications (0975 – 8887) Volume 87 – No.16, February 2014.

[5] Jenna Carr, (2014), "An Introduction to Genetic Algorithm".

[6] Jyoti G. Borade, (2013), "Software Project Effort and Cost Estimation Techniques" International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, pp 730-739.

[7] Simon, D. (2013). Evolutionary Optimization Algorithms: Biologically-Inspired and Population-Based Approaches to Computer Intelligence. Hoboken: Wiley.

[8] K. Subba Rao, et. al. (2013), "Software Cost Estimation in Multilayer Feed forward Network using Random Holdback Method", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 10, pp 1309-1328.

[9] Matlab 2012a: Optimization Toolbox-Product Documentation.

[10] Vahid Khatibi, Dayang N. A. Jawawi, (2011), "Software Cost Estimation Methods: A Review", Journal of Emerging Trends in Computing and Information Sciences, Volume 2 No. 1, pp 21-29.

[11] Erik D. Goodman, (2009), "Introduction to Genetic Algorithms", 2009 World Summit on Genetic and Evolutionary Computation Shanghai, China. [12] Stein Grimstad, et. al, (2006), "A Framework for the Analysis of Software Cost Estimation Accuracy", ISESE'06, ACM 1-59593-218-6/06/0009.

[13] Andrey Popov, (2005), "User Manual, Genetic Algorithm for Optimization", Programs for MATLAB, ver. 1.0.

[14] Mitchell, M. (1996). An Introduction to Genetic Algorithms. Cambridge: MIT Press.

[15] Bernard L. "Cost Estimation For Software Development", Addision_Wesley, 1987.

[16] Albrechet, A.J. etc. "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation", IEEE on Software Engineering, NOV 1983.

[17] Boehm, B.W. "Software Engineering Economics", Prentice_Hall, 1981.