RESEARCHERID
THOMSON REUTERS

[*Sangeetha* et al., 4(6): June, 2017*] ISSN: 234-5197
Impact Factor: 2.715

INTERNATIONAL JOURNAL OF RESEARCH SCIENCE & MANAGEMENT

SOFTWARE REFACTORING COST ESTIMATION USING PARTICLE SWARM OPTIMIZATION

M. Sangeetha*, Dr. P. Sengottuvelan
*Ph.D Research scholar, Department of computer science, Periyar University PG Extension Centre, Dharmapuri - 636705, INDIA
Associate Professor, Department of computer science, Periyar University PG Extension Centre, Dharmapuri - 636705, INDIA

DOI: 10.5281/zenodo.583651

Keywords: cost estimation, PSO, COCOMO, estimate PSO.

Abstract
"Now a day's software industry major development cost devoted software maintenance. The major challenge for this industry is to produce quality software which is timely designed and build with proper cost estimates. Refactoring is used increasing the ability of software to adopt the new requirements and maintenance In this paper, we have proposed a cost estimation model based on Multi-objective Particle Swarm Optimization (MPSO) to tune the parameters of the famous Constructive Cost Model (COCOMO). This cost estimation model is integrated with Quality Function Deployment (QFD) methodology to support decision making in software designing and development processes for improving the quality. This approach helps to the developers to efficiently plan the overall software development life cycle.

Introduction
In software industrial field, the software cost estimation has also emerged as a major issue. The cost predict for any software product is the most hard task which is penetrating for its customers, developers and users. It affects the total software project development process including contract negotiations, scheduling, resource allocation and project planning [1, 2]. Specious cost estimation and poor code design may result in complete software failure. Over estimation and code drift may result in low quality software and late delivery. Thus a accurate cost estimation is always needed to correctly set up the software development [2] and to avoid the harmful market consequences arising because of missing the deadlines and producing low quality products [3, 4].

Many of the proposed cost estimation models such as Putnam's SLIM [5], RCA's PRICE-S [6] and Boehm's COCOMO II [4] depends on the size of the software. Some other non-parametric models include expert judgments; break down structures, regression based and ynamics based models [7]. Recently, many researchers have explore the domain of evolutionary working out techniques [8,9, 10] to form better opinion models [3, 13, 14,15, 16] because of the remarkable exploration and optimization capabilities of these algorithms [9,10] and their power to tackle ambiguity and suspicions [15, 16]. In this paper also we have proposed one of these techniques. We have tried to scrutinize Particle Swarm Optimization (PSO)[11, 12], which is a pretending optimization technique based on the movement and intelligence of swarms. Here PSO is used to build a more perfect cost estimation model, by alteration the COCOMO model parameters.. It is an efficient study to establish a relation between. The aim of QFD is to transform subjective quality criteria into quantifiable and measurable objective ones which can then be used in the designing and developing the project resulting in increased product acceptance. In the rest of the paper is organized as follows: First we have described the QFD technique then the application of QFD in guiding the software process is detailed. Secondly software cost estimation problem is described with a COCOMO based mathematical model and finally a solution is proposed through PSO.

Software Cost Estimation Procedure
Refactoring is reengineering within the object oriented context. Software refactoring can be defined as "the process of changing a software system in such a way that it does not alter the external behavior of the code yet

improves its internal structure". Refactoring if applied on the working software than it ameliorates the performance of the support contrasting with the principal of "If Its Working Don't Change". Refactoring can be applied on the poorly working program which contains some bugs that are to be fixed. If you have a poorly factored program that does what the customer needs and has no serious bugs, then you may feel not to apply refactoring on it. When you need to fix a bug or add a feature, you refactor such as Extreme Programming has generated a great amount of interest in refactoring, and refactoring support has become a required feature in modern-day IDEs. The key insight is that it's easier to rearrange the code correctly if you don't simultaneously try to change its functionality. The existing tools provide only the feature of transformation from existing to the new design where as the proposed model in this paper can be used for the calculation of the cost required to perform refactoring that we have proposed five opportunities to be refactored.

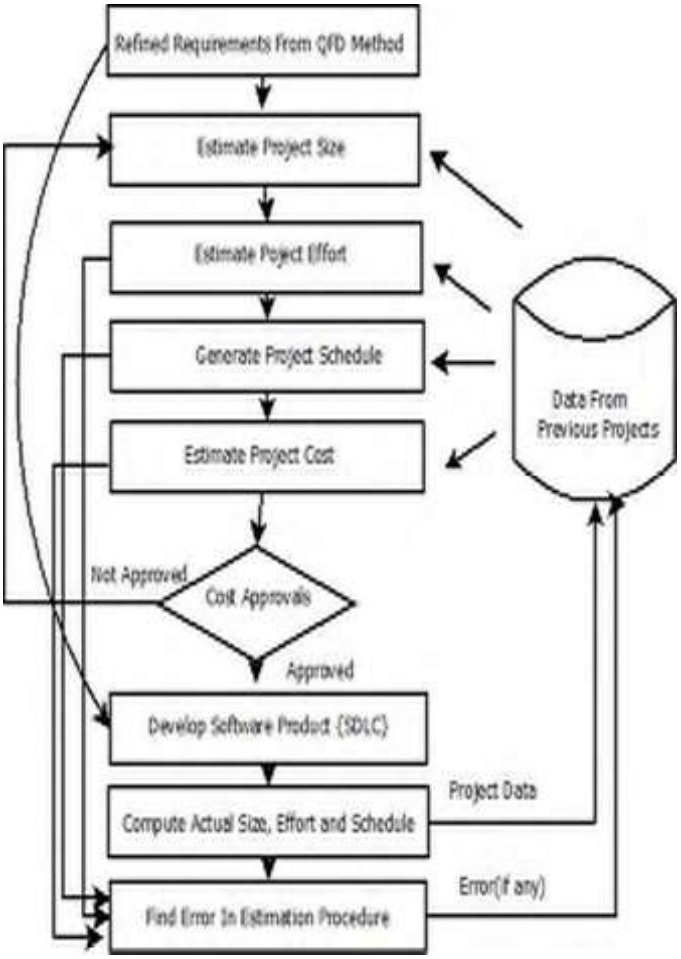

Figure 1: Software project estimation procedure

Particle Swarm Optimization (PSO)

PSO is a population based computational technique that optimizes a given problem with the help of particles that moves in the search space [11].Every particle retains a track of its locations in the problem space. These locations/coordinates are associated with the best solution it has reached so far. This best solution is known as *pbest*. Another best value that is recorded by the particle swarm optimizer is the best value, obtained so far by any particle in the population is called *gbest*. The movement/direction of movement of any particle is controlled by these two values: *pbest* and *gbest* [12]. The PSO technique is described in algorithm

Why PSO?

It is highly desirable to get the accurate estimates of cost and effort, but no prototype has proved to be effective at efficiently and reliably predicting software development cost because of the uncertainties, contingencies and imprecisions associated with the software development process [35]. Due to these characteristics the focus of researchers has been shifted to use the natural computing paradigm models that simulate procedures to learn and react in an uncertain environment. PSO is one of these optimization techniques, which is both robust and stochastic. It is based on the methodology to solve a new problem by adapting the solutions of previous similar problems. Also PSO is fast and cheap and since its evolution PSO has been successfully deployed in various research and application areas ranging from engineering utilities to business critical software to real time optimization procedures [36]. Thus PSO is chosen as an ideal algorithm to solve the software effort estimation problem.

Algorithm 2: Particle Swarm Optimization (PSO)

Given:
Vki= Velocity of ith particle at kth iteration.
Sik= Position of ith particle at kth iteration.
pbest = pbest of ith particle.
Gbest = gbest of the whole population.
w, c1 and c2 are weighting variables and rand is uniformly distributed random number generator between 0 and 1.

$V_i^{k+1} = {}^{w}V_i^k +$ G.rand()$(pbase_i - S_i^k) + C_2\, rand_3(\quad) gbase_1^{k-} S_i^k$

$$S_i^{k+1} = S_1^k + V_i^{k+1}$$

Step1. Initialize all the particles with random positions and velocities.

Step2. For all particles
a) Evaluate their fitness *f* on the basis of their location *l*.
b) If *f* is better then *pbest* then *pbest= l.*

Step3. Set best of *pbest* as *gbest*.

Step4. Update each particle's velocity and position according to equation (4) and (5).

Step 5. GOTO 2 until maximum iterations.

Step 6. Return *gbest* as optimal solution

Estimation using PSO

In our model, we have used PSO to calibrate the COCOMO model parameters. This model would be responsible for optimizing the estimated effort. students and the work groups The students are divided groups of 5 to 6. The groups were formed based upon the academic performance of the previous terms as well as the previous software development experience. Each group contains 2-3 people and they all had two years of experience in the field of software development. Table1 contains the previous knowledge and experiences that the students had prior to the beginning of the software project.

Table 1. Previous knowledge and experiences

Characteristics	Experience and knowledge
Project Managemer	Short and medium software programming Projects
Programming Platfc	C, C++, Java, C#, Python
Databases	SQL server, MS SQL server
Analysis and Design	Object oriented design
Software Estimation	No previous experience

The PSO is followed by each group the implementation each group. Every group had to develop two iterations in the implementation phase. Each iteration had taken two weeks of work. The number of use cases implemented in the second and third iteration was based on the previous iterations. There were four use cases that were implemented for each iteration.

Table 2. Software Used for the Projects

Item	Software
Documenting Softwar	MS office suite 2007/201(
Modeling Software	Rational Rose
Programming Languag	C++, Java
Data base	Microsoft SQL server 200
Operating System	Linux, Unix

Experimental Results

The actual efforts are measured in men-hours. The actual effort is determine by the effective work done by each group for software design, testing and implementation

. Table 3: Actual Effort for Group A Project

Iteration	Group A	
	Actual Effort	UFP
1	292	187.23
2	189	195.61

Table 4.: Actual Effort for Group A Project

Iteration	Group B	
	Actual Effort	UFP
1	322	155.13
2	277	178.30

Magnitude of Relative Error

A Magnitude of relative error is showing the deviation between the prediction of the formula and the observed data.

Magnitude of Relative Error = $actual\ effort - Estimation\ effort / actual\ effort$

Table 4 Magnitude of relative error: Group A

Estimated Effort	Iteration	EAF Factor	Actual Effort	MRE
1.845	1	1.46	1.753	9.2%
1.585	2	0.8	1.544	2.65%

Table 5 Magnitude of relative error: Group B

Iteration	EAF Factor	Actual Effort	Estimated Effort	MRE
1	1.46	1.932	1.687	12.6%
2	0.8	1.143	1.278	11.8%

Table 6 Magnitude of relative error: Group C

teration	EAF Factor	Actual Effort	Estimated Effort	MRE
1	1.46	1.738	1.952	12.31%
2	0.8	0.893	0.972	8.8%

Table 7 Magnitude of relative error: Group

Iteratic	EAF Factor	Actual Effort	Estimated Effort	MRE
1	1.46	1.738	1.952	12.31%
2	0.8	0.893	0.972	8.8%

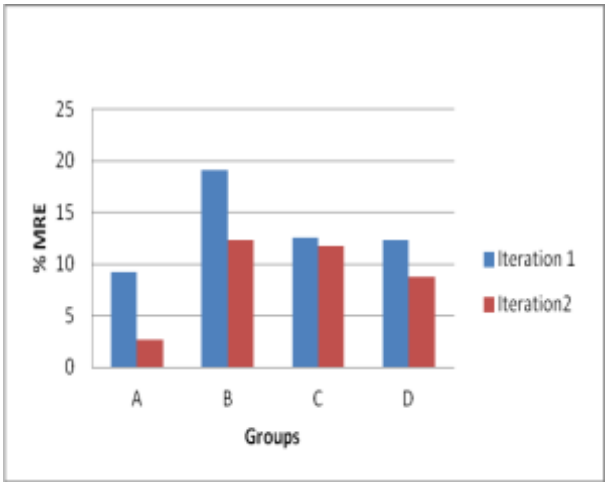

Fig 2 Magnitude of relative error in consecutive iteration

Conclusion

Software cost estimation is based on a probabilistic model and hence it does not generate exact values. However availability of good historical data coupled with a systematic technique can generate better results. In this paper we proposed new model structure to estimate the We have used PSO to build a suitable model for software cost estimation by tuning the COCOMO parameters. We have also used QFD technique to establish a high correlation between customer requirements and design specification. This not only guides the software development process but helps to find the cost driver's values also. Thus gives a better cost estimate. So, this paper is an integration of QFD technique and PSO method to develop a more precise software cost estimation model

References

[1] T. Benala, S. Dehuri, S. Satapathy and S. Raghavi, "Genetic Algorithm for Optimizing Neural Network Based Software Cost Estimation", Lecture Notes in Computer Science, Springer Berlin, vol. 7076, pp. 233- 239, 2011.
[2] H. Leung, Z. Fan, "Software Cost Estimation", Handbook of Software Engineering, Hong Kong Polytechnic University, 2002.
[3] A. Sheta, D. Rine and A. Ayesh, "Development of Software Effort and Schedule Estimation Models Using Soft Computing Techniques", IEEE Congress on Evolutionary Computation, 2008.
[4] B. W. Boehm, C. Abts, et al. "Software Cost Estimation with COCOMO II", Upper Saddle River, NJ, Prentice Hall PTR, 2000.
[5] L. H. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem", IEEE Transactions on Software Engineering 4, pp. 345-361, 1978.

[6] F. R. Freiman, R.E. Park, "PRICE Software Model – Version 3: An Overview", IEEE-PINY Workshop on Quantitative Software Models, pp. 32–41, 1979.
[7] Y. F. Li, M. Xie and T. N. Goh, "A Study Of Project Selection Feature Weighting For Analogy Based Software Cost Estimation", The Journal Of Systems and Software 82, pp. 241–252, 2009.
[8] C. A. CoelloCoello, "Evolutionary Multiobjective Optimization: A Historical View of the Field", IEEE computational Intelligence Magazine, 1, pages 28-36, 2006.
[9] K. Deb, "Multi-Objective Optimization Using Evolutionay Algorithms", John Wiley, Chichester, UK (2001).
[10] EckartZitzler, Marco Laumanns and Stefan Bleuler, "A Tutorial on Evolutionary Multiobjective Optimization, Metaheuristics for Multiobjectiv eOptimisation", Springer. Lecture Notes in Economics and Mathematical Systems Vol. 535, Berlin, pp. 3-37, 2004.
[11] J. Kennedy and R. Eberhart, "Particle swarm optimization", in Proc. IEEE Int. Conf. Neural Networks, pp. 1942–1948, 1995.
[12] R. C. Eberhart and Y. Shi, "Particle swarm optimization: Developments, applications and resources," in Proc. IEEE Int. Conf. Evolutionary Computation, vol. 1, pp. 81–86, 2001.
[13] Tad Gonsalves, Atsushi Ito, Ryo Kawabata and Kiyoshi Itoh, "Swarm Intelligence in the Optimization of Software Development Project Schedule", in Proc. IEEE, 0730-3157/08 , 2008.
[14] J .S. Pahariya, V. Ravi and M. Carr, "Software Cost Estimation using Computational Intelligence Techniques", World Congress on Nature and Biologically Inspired Computing, 2009.
[15] M.W. Nisar, J. W. Yong and M. Elahi, "Software development effort estimationusing fuzzy logic - A survey", IEEE International Conference on FuzzySystems and Knowledge Discovery, vol.1, pp: 421-427, 2008.
[16] D. Kumar, D. Kashyap, K. K. Mishra and A. K. Misra, "Security Vs cost: An issue of multi- objective optimization for choosing PGP algorithms", IEEE ICCCT-2010, India, 2010