Indian Journal of Science and Technology, *Vol 9(31), DOI: 10.17485/ijst/2016/v9i31/93058, August 2016*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Software Development Cost Estimation: A Survey

Samina Rajper*[*] and Zubair A. Shaikh

Department of Computer Science, Shah Abdul Latif University Khairpur, Sindh, Pakistan;
samina.rajper@gmail.com, zubair.shaikh@jinnah.edu

Abstract

Objectives: The present study is undertaken to survey the Software development cost estimation techniques. This study will provide guidelines and for researchers and practitioners of software engineering. **Methods/Analysis:** The study was undertaken by planning, conducting and reporting the literature review (LR) for the years 1991-2016. **Findings:** The study revealed that several SDCE models have been introduced. The reason for the evolution of software cost estimation models may be the changing nature of software complexity, i.e., one cannot exactly predict the cost for the whole project. Not only conventional empirical and quantitative methods but several data mining and machine learning techniques are also used for improved results. However, it is revealed that from quantitative to empirical all SDCE models can be used alone or hybrid with robust ML or DM techniques to estimate the software development exertion.

Keywords: COCOMO, Data Mining Techniques, Machine Learning Techniques, Software Development Cost Estimation

1. Introduction

Software Development Cost Estimation (SDCE) is the procedure of foreseeing the exertion required to build up a software product/system. By and large speaking, SDCE can be considered as a sub-area of Software engineering, which incorporates the forecasts software development as well as its maintenance cost estimation. It is thought to be the foundation for project bidding, budgeting and planning. Cost estimation and good predicting results the smoother process throughout the project. Different Software cost estimation techniques have been emerged in last three decades[1]. SDCE models are used for different purposes, i.e., trade off, Budgeting, risk analysis, Planning and control and investment analysis for software improvement. Since 1980s, numerous estimation techniques have been proposed in SDCE space. The main focus of these models was software complexity estimation in terms of man-effort and codes' lines calculation. Many research studies have been undertaken to survey various estimation techniques, i.e., Jørgensen and Shepperd[2] identified 11 SDCE techniques, research study[1] identifies eight techniques and several others will be discussed in later sections.

This study is undertaken to comprehensively survey the SDCE techniques. The objective of the study is to explore various techniques used for SDCE for researchers and practitioners.

2. Materials and Methods

The study was undertaken by planning, conducting and reporting the literature review (LR) for the years 1991-2016. Literature review was conducted by selecting the good reference research studies published in best journals. The objective of the study is to explore various techniques and models for SDCE for researcher and practitioners. The next section presents the findings of the study. The method to conduct this literature survey is shown by Figure 1.

3. Results and Discussions

The study revealed that several SDCE models have been introduced. The reason for the evolution of software cost estimation models may be the changing nature of software complexity, i.e., one cannot exactly predict the cost for the whole project. SDCE includes the determination

*Author for correspondence

Figure 1. Workflow of the Study.

of the effort in terms of human-months, project duration time and total cost in dollars[3]. These SDCE models are either empirical Models which use the previous projects' data for current projects' evaluation and estimation, i.e., COCOMO[4,5] while some are analytical Models which use formulas to estimate, i.e.,[6-8]. Moreover, the SDCE models can be divided into various families, i.e., Model based, Expertise based, Machine learning oriented, Dynamics based, Regression based and composite based[9]

3.1 Model-based

In the last decades, we have various proprietary models. Therefore it is not possible to compare them as model. Following is an overview of some of these models:

3.1.1 Putnam's Software Life-cycle Model or (SLIM)

SLIM (Software Life-cycle Model) was introduced by Larry Putnam in 1970s who belonged to Quantitative SDCE Model paradigm. The foundation of this model was Putnam's analysis of Life cycle[5,6,9]. The model was actually based on Rayleigh distribution for project personnel level in comparison of time[5]. The shape of the curve is affected by Man power Buildup Index (MBI) and Productivity factor (PF). It is one of the quality of this model that the data can be recorded and analyzed from past projects. In case of unavailability of data then a set of questions can be satisfied to acquire the resultant values of MBI and PF from the database[9].

3.1.2 Checkpoint

It was introduced by Vapers Jones and its foundation is knowledge-based software project estimation in 1980's[9,10]. Function/Feature points were used as its basic inputs and the main focus was on the areas, i.e., estimation, measurement and Assessment.

3.2 Expert based Model

In absence of any empirical data, the relevant personnel and experts are captured to predict cost estimation on the basis of their experience and past projects lessons. The common techniques used are:

3.2.1 Delphi Technique[5,9,11]

3.2.2 Work Breakdown Structure (WBS)[9,12]

3.3 Learning-Oriented Techniques and Hybrid Techniques

Learning-oriented technique (LOT)/Machine Learning Techniques (MLT) are based on some of the oldest and some of the newest techniques applied for the estimation[9], i.e., Case Studies, Neural Networks etc. It is mentioned by[1] eight different MLT used for SDCE from 1991-2010, i.e., Support Vector Regression (SVR), Artificial Neural Networks (ANN), Bayesian Networks (BN), Genetic Programming (GP), Association Rules (AR), Genetic Algorithms (GA), Case-Based Reasoning (CBR), Decision Trees (DT). While[13] used a hybrid approach with neural network and genetic algorithm and[14] used ant colony and chaos optimization also[15] used Hybrid approach of Particle Swarm Optimization with Fuzzy C-means and Learning Automata in SDCE.

3.4 Composite Techniques

On the basis of the fact that each and every technique defined above has many pros and cons. Therefore, researchers introduced techniques which are called composite techniques because they incorporate two or more techniques. One of the technique is Bayesian Approach which proved to be the base of the development of the COCOMO II model[9]. COCOMO stands for Constructive Cost Model, which was first published by Barry Boehm[5]. However, several cost estimation model for software products have been introduced but the dominant one is COCOMO. The model can be explained as three levelled model, i.e.,

- ***3.4.1*** ***COCOMO 815*** is a single-valued and static model, is the basic model[9]. This model is capable of computing the effort and takes the software cost as the function of program size whereas the program size can be several line of code[16]
- ***3.4.2*** ***The COCOMO Intermediate*** is capable of computing software development efforts in a very systematic way as it deals with the development effort of system as a function of program size and set of 15 cost drivers[17,14].
- ***3.4.3*** ***Detailed COCOMO*** is capable of not only computing all defined drivers by Intermediate COCOMO but also can be capable of assessing each cost driver's effect on each phase[9,17].

By the pace of time, many research studies have been undertaken to extend the COCOMO suit. Figure 2 is used to represent the historical evolution of COCOMO suit. Following table is used to understand the pros and cons of various types of techniques for software cost estimation. In the light of these facts and figures, it is obvious that why a continuous evolution for any model is important.

? is used to mention no any factor is identified1

3.5 SDCE using Data Mining Techniques

Boehm's COCOMO[4,5,18] is the most popular SDCE model but in recent years many Data Mining techniques are also used for SDCE. In this section some of the research studies will be represented, i.e.,[19] has used Artificial Neural

Table 1. Shows different activities covering by various SDCE Models1

Group	Factor	Checkpoint	SLIM	ESTIMACS	PRICE-S	COCOMO II
Program Attributes	Type/Domain	✓	✓	✓	✓	✓
	Complexity	✓	✓	✓	✓	✓
	Language	✓	✓	?	✓	✓
	Reuse	✓	✓	?	✓	✓
	Required Reliability	?	?	✓	✓	✓
Computer Attributes	Resource Constraints	?	✓	?	✓	✓
	Platform Volatility	?	?	?	?	✓
Personnel Attributes	Personnel Capability	✓	✓	✓	✓	✓
	Personnel Continuity	?	?	?	?	✓
	Experience	✓	✓	✓	✓	✓
Project Attributes	Tools and Techniques	✓	✓	✓	✓	✓
	Breakage	✓	✓	✓	✓	✓
	Schedule	✓	✓	✓	✓	✓
	Process Maturity	✓	?	?	?	✓
	Team Cohesion	✓	?	?	✓	✓
	Security Issues	?	✓	?	?	✓
	Project	✓	✓	✓	✓	✓
Activities Covered	Inception	✓	✓	✓	✓	✓
	Elaboration	✓	✓	✓	✓	✓
	Construction	✓	✓	✓	✓	✓
	Transition and Maintenance	✓	✓	X	✓	✓
Size Attributes	Source-Instructions	✓	✓	X	✓	✓
	Function- Points	✓	✓	✓	✓	✓
	OO-related metrics	✓	✓	?	✓	✓

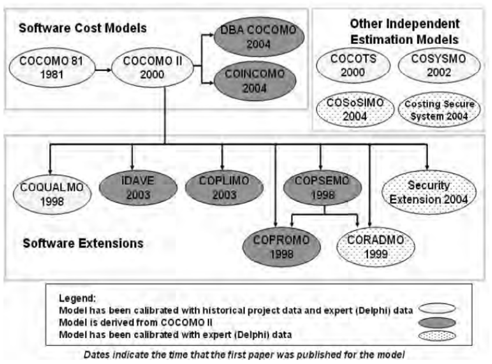

Figure 2. Historical evolution of COCOMO suit [18].

Network (ANN), Linear Regression (LR), K-Nearest Neighbours (KNN) and Support Vector Regression (SVR) to compare different data mining techniques.[20] used ordinary least square regression (OLSR), case-based reasoning (CBR) technique for SDCE.[21] used CBR, CART, OLSR and ANOVA techniques for SDCE.[22-24] also used data mining techniques to improve SDCE method.[25] identified several data mining techniques used in various studies[26-31] Least median squares regression, Model tree, MARS, Multilayered perceptron neural network, Radial basis function networks, Least squares support vector machines. Robust regression, OLS regression with log transformation, least squares regression (OLSR), OLSR with Box Cox (BC) transformation, Ridge regression, Case-based reasoning, CART.

4. Conclusion

This study identified various techniques belonging to various domains for SDCE. Some techniques were used as hybrid to improve the already famous models. It is found that the development environment is continuously evolving and rapidly changing in nature. Several factors of software development process may be inter-related to each other therefore, anyone technique will not be said the most appropriate or suitable for SDCE. However, it is revealed that from quantitative to empirical all SDCE models can be used alone or hybrid with robust ML or DM techniques to estimate the software development exertion.

5. References

1. Wen J, Li S, Lin Z, Hu Y, Huang C. Systematic literature review of machine learning based software development effort estimation models. Information and Software Technology. 2012; 54(1):41–59.
2. Jorgensen M, Shepperd M. A systematic review of software development cost estimation studies. IEEE Transactions on Software Engineering. 2007; 33(1):33–53.
3. Leung H, Fan Z. Software cost estimation. Handbook of Software Engineering, Hong Kong Polytechnic University. 2002; 1–14.
4. Boehm BW, Madachy R, Steece B. Software cost estimation with Cocomo II with Cdrom: Prentice Hall PTR, 2000.
5. Boehm BW. Software engineering economics: Prentice-hall Englewood Cliffs (NJ), 1981.
6. Putnam LH. A general empirical solution to the macro software sizing and estimating problem. IEEE Transactions on Software Engineering. 1978; 4(4):345.
7. Parr FN. An alternative to the Rayleigh curve model for software development effort. IEEE Transactions on Software Engineering. 1980; (6):291–6.
8. Cantone G, Cimitile A, De Carlini U. A comparison of models for software cost estimation and management of software projects. Computer systems: performance and simulation. 1986; 123–40.
9. Boehm B, Abts C, Chulani S. Software development cost estimation approaches—A survey. Annals of software Engineering. 2000; 10(1-4):177–205.
10. Boehm BW, Valerdi R. Achievements and challenges in cocomo-based software resource estimation. Software, IEEE. 2008; 25(5):74–83.
11. Woudenberg F. An evaluation of Delphi. Technological forecasting and social change. 1991; 40(2):131–50.
12. Tausworthe RC. The work breakdown structure in software project management. Journal of Systems and Software. 1979; 1:181–6.
13. Molani M, Ghaffari A, Jafarian A. A new approach to software project cost estimation using a hybrid model of radial basis function neural network and genetic algorithm. Indian Journal of Science and Technology. 2014; 7(6):838–43.
14. Dizaji ZA, Gharehchopogh FS. A hybrid of ant colony optimization and chaos optimization algorithms approach for software cost estimation. Indian Journal of Science and Technology. 2015; 8(2):128–33.
15. Gharehchopogh FS, Ebrahimi L, Maleki I, Gourabi SJ. A Novel PSO based approach with hybrid of Fuzzy C-means and learning automata in software cost estimation. Indian Journal of Science and Technology. 2014; 7(6):795–803.
16. Abbas SA, Lar SU, Liao X, Naseem RA. Software Models, Extensions and Independent Models in Cocomo Suite: A Review. Journal of Emerging Trends in Computing and Information Sciences. 2012; 3(5):1–11.
17. Boehm B. Making RAD work for your project. Computer. 1999; 32(3):113–4, 7.

18. Boehm B, Valerdi R, Lane J, Brown A. COCOMO suite methodology and evolution. CrossTalk. 2005; 18(4):20–5.
19. Khalifelu ZA, Gharehchopogh FS. Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation. Procedia Technology. 2012; 1:65–71.
20. Finnie GR, Wittig GE, Desharnais J-M. A comparison of software effort estimation techniques: using function points with neural networks, case-based reasoning and regression models. Journal of Systems and Software. 1997; 39(3):281–9.
21. Briand LC, Langley T, Wieczorek I, editors. A replicated assessment and comparison of common software cost modeling techniques. Proceedings of the 22nd International Conference on Software Engineering, Limerick. 2000. p. 377–86.
22. Sajadfar N, Ma Y. A hybrid cost estimation framework based on feature-oriented data mining approach. Advanced Engineering Informatics. 2015; 29(3):633–47.
23. Ebrahimpour N, Gharehchopogh FS, Khalifehlou ZA. A New Approach with Hybrid of Artificial Neural Network and Ant Colony Optimization in Software Cost Estimation. MAGNT Research Report. 2015.
24. Gharehchopogh FS, Pourali A. A new approach based on continuous genetic algorithm in software cost estimation. Journal of Scientific Research and Development. 2015; 2(4):87–94.
25. Dejaeger K, Verbeke W, Martens D, Baesens B. Data mining techniques for software effort estimation: a comparative study. IEEE Transactions on Software Engineering. 2012; 38(2):375–97.
26. Chiu N-H, Huang S-J. The adjusted analogy-based software effort estimation based on similarity distances. Journal of Systems and Software. 2007; 80(4):628–40.
27. Park H, Baek S. An empirical validation of a neural network model for software effort estimation. Expert Systems with Applications. 2008; 35(3):929–37.
28. Kumar KV, Ravi V, Carr M, Kiran NR. Software development cost estimation using wavelet neural networks. Journal of Systems and Software. 2008; 81(11):1853–67.
29. Li Y-F, Xie M, Goh TN. A study of project selection and feature weighting for analogy based software cost estimation. Journal of Systems and Software. 2009; 82(2):241–52.
30. Strike K, El Emam K, Madhavji N. Software cost estimation with incomplete data. IEEE Transactions on Software Engineering. 2001; 27(10):890–908.
31. Li J, Ruhe G, editors. A comparative study of attribute weighting heuristics for effort estimation by analogy. Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering, New York. 2006; 66–74.