Using Simulation to Determine the Sensibility of Error Sources for Software Effort Estimation Models

Mercedes Ruiz[1], Juan-José Cuadrado Gallego[2], Miguel-Angel Sicilia[2], Daniel Rodríguez[3]

[1]The University of Cádiz, Spain
mercedes.ruiz@uca.es

[2]The University of Alcalá, Spain
jjcg@uah.es, msicilia@uah.es

[3]The University of Reading, Reading, RG6 6AY, United Kingdom
d.rodriguezgarcia@reading.ac.u

Abstract:

In this paper, a software project simulator based on System Dynamics is used to analyze the different sources of error in the initial estimates of parametric models. More precisely, the effect of using the two different COCOMO models, different calibration models, and a different selection of ratings for the effort multipliers in estimates is shown. The bdel-Hamid' system dynamics simulation model is used as a technique to observe the potential evolution of each error source during the life-cycle and its effects on the key variables of the project such as effort, productivity, schedule, etc. The study determines which of the three sources introduces a higher error in the initial estimations and also their effects in a project the end of the project. We observed how simulation helped to minimize the effect of these errors, corroborating the need for multiple estimation methods.

Kewords

Parametric software effort estimation, simulation modeling, COCOMO II

1 Introduction

Estimation based of expert judgement Parametric software cost estimation models are one of the principal effort estimation methods used in software project management (Boehm, Abts and Chulani, 2000). Parametric estimation is based on using historical project databases and expert knowledge to derive a Cost Estimation Relationship (CER) that relates the effort spent to one or several cost drivers that are known to affect the development process. Within this type of cost estimation models, three major aspects can be considered to determine the quality of the results:

(a) The CER expression to be used, not only regarding the mathematical model selected, but also regarding the input variables selected. A considerable amount of research on parametric estimation models has focused on this aspect such as [1-4]

(b) How models are calibrated. There are also many calibration studies regarding this aspect [5-7].

(c) The methods used to select the values for the cost drivers for each specific project. This aspect has attracted less attention. Baik and Boehm [8] described the decomposition of the COCOMO TOOL cost driver, which was later applied as a technique to improve predictive accuracy. Similarly, Cuadrado *et al* [9, 10] described the analytical decomposition of the COCOMO DOCU variable. Other studies such as [11] show that the correct assessment of the cost drivers inputs for each particular project is acknowledged as a important milestone in parametric estimation.

In this paper, System Dynamics simulation techniques [12] based on the Abdel-Hamid and Madnick model [13] are applied to different versions of the original COCOMO [13] and COCOMO II postarchitecture models [4], in an attempt to study the sensibility of each of the three above mentioned error sources. Results point out that the relative importance of (c) as an error source is at least similar to the remaining two. Evidence is provided not only for the estimations carried out at the beginning of the project, but also through the different phases of project development.

The rest of the paper is structured as follows. Section 2 gives an overview of software project simulators and the structure of the model used. The research method and the assumptions taken in this study are detailed in Section 3. Section 4 collects the results offered by using the parametric models and the final simulation results. Finally, Section 5 draws the conclusions and further work.

2 Related work

2.1 Parametric Cost Estimation models and COCOMO

The original COCOMO (COnstructive COst MOdel) [13], also known as COCOMO 81, models predict total effort that would be necessary to generate a source program of certain complexity. The prediction is represented by the equation: $eff = c\ L^{b}$ where *eff* is the effort in person-months, L is the code size in *KLOC*, c and *b* are constants, dependent on the complexity of the problem (organic, semi-detached or embedded). The intermediate and detailed COCOMO models make plans (effort and schedule estimates) for each development phase

of the standard software project using a set of so-called cost driver attributes (with associated effort multipliers). The detailed COCOMO model differs from the intermediate COCOMO model in that it uses different effort multipliers for each phase of a project. The phase dependent effort multipliers yield better estimates than the intermediate model.

In its latest version, COCOMO II [14] acknowledges that the numbers of lines of code are unknown at early stages of the development process. As a result, COCOMO II divides the estimation process into 3 stages: (i) Stage 1, Application Composition, where size estimates are performed in object points; (ii) Stage 2, Early Design to explore alternative architectures and concepts. Size estimates are done in function points; and (iii) Stage 3, Post-architecture when development has begun estimation uses lines of code as in COCOMO 81.

2.2 System dynamics

System Dynamics [12] is a method for studying how complex systems change over time where internal feedback loops within the structure of the system influence the entire system behaviour. The application of system dynamics to software projects provides the perspective of considering them as complex socio-technological dynamic systems, whose evolution will be determined by their internal structure as well as the relations established inside the working team. This allows the development of dynamic models to describe the feedback structure of the system being modelled as well as the mental process followed by project managers in making decisions. It is to be noted that decision making has been traditionally based on a manager's experience. The building of a dynamic model for a software project is based on the evolution of the project. Therefore, the attainment of the project goals such as meeting the deadlines, a project phase being within budget etc. depends on: (i) the initial estimations of the necessary resources; (ii) the management policies to be applied; (iii) the characteristics of project (number of tasks, time, cost, number of technicians, software complexity, etc.); and (iv) the characteristics of the organisation (maturity level, average delay, turnover on the project's work force, etc).

The use of simulation environments (such as Stella, Vensim, iThink, Powersim, etc.) for software projects in the beginning of the 90s paved the way for tools which could allow us to simulate the behaviour of the projects. With these software project simulation tools, project managers could experiment different management policies without additional cost [3][6]. A software project simulator allows us to analyse the following:

1. A priori project analysis, to simulate the project before initiation.
2. Project monitoring, to simulate the project during its development phase for adapting the project estimation to its actual evolution.
3. Post-mortem analysis through the simulation of a finished project, to know how the results obtained could have been improved.

In summary, software projects simulators allow us to answer questions such as: "what would happen if...?" before beginning; "what is happening...?" during the execution and "what would have happened if...?" once the project is finished.

Abdel-Hamid and Madnick developed a model in order to study the effects of management policies and actions on software development projects. Developed to help understanding the software development process, it allows us to evaluate the impact of management policies. Being the software development of complex systems difficult to understand in its entirety, Abdel-Hamid's model is partitioned into four subsystems that are manageable and understandable (see Figure 1):

1. Human Resource Management Subsystem deals with hiring, training, assimilation and transfer of the human resources.
2. Software Production Subsystem models the software development process excluding requirements, operation and maintenance. This subsystem has four sectors: Manpower Allocation, Software Development, Quality Assurance and Rework, and System Testing.
3. Control Subsystem deals with the information that decision makers have available. The model takes into account that it is difficult to know the true-state of a process during development as usually such information is inaccurate.
4. Planning Subsystem takes into account the initial project estimates. Such estimates need to be revised through the software project life cycle.

Other authors have combined Systems Dynamics and COCOMO as a way to calibrate or verify simulation models. Madachy [15] created a model to examine the effects of inspection practices on cost, schedule and quality thought the lifecycle and it was calibrated using COCOMO

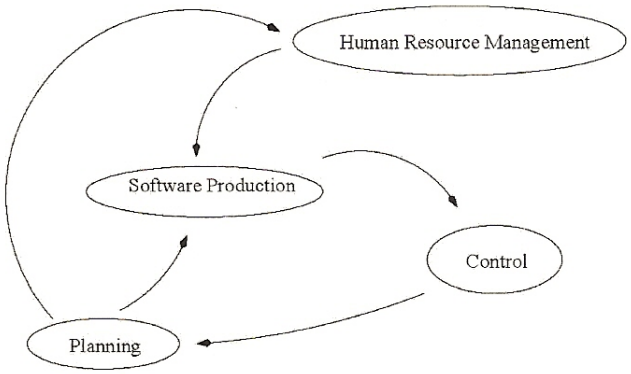

Figure 1: Abdel-Hamid's Model

3 Method and assumptions

The study is divided in three parts, one for each of the sources of error described in the introduction. For the three, a common methodological scheme was used. First, a comparison of error estimations were computed, in an attempt to find evidence about each particular source of error. After that, the simulation model is used as a technique to observe the potential evolution of each error source. Finally, the pre- and post-simulation assessments are examined to compare the impact of each of the error sources.

Regarding the error source (a), which is the CER expression to be used, the impact of the model is study by comparing the CERs used in the COCOMO 81 [13] (1981 calibration, and considering *EMi=1* for all the effort multipliers) and COCOMO II (post-architecture, 1997 calibration, also with *EMi=1* for all the effort multipliers) versions.

The study of error source (b), different calibrations for a single estimation model, we used the COCOMO II and its calibrations corresponding to the years 1997 and 1998. In this case, the outputs of the different calibrations when two effort multipliers (CPLX and DATA) take high values are analyzed.

For the study of the error source (c), selection of the values for the cost drivers for each specific project, a comparative analysis of the differences in possible rating values for a number of cost drivers was carried out. Concretely, the rating values and effort multipliers used are provided in Table 1.

Table 1: Selection cases used in the study of error source (c)

Case	*Selection 1*	*Selection 2*	*Model used*
1	EM(CPLX) = 1,15 (HIGH), EM(DATA) = 1,08 (HIGH)	EM (CPLX) = 1,30 (VERY HIGH), EM(DATA) = 1,16 (VERY HIGH)	COCOMO81
2	EM(CPLX) = 1,15 (HIGH), EM(DATA) = 1,09 (HIGH)	EM(CPLX) = 1,30 (VERY HIGH), EM(DATA) = 1,19 (VERY HIGH)	COCOMO II Postarchitecture, 1997 calibration.
3	EM(CPLX) = 1,17 (HIGH), EM(DATA) = 1,14 (HIGH)	EM(CPLX) = 1,34 (VERY HIGH), EM(DATA) = 1,28 (VERY HIGH)	COCOMO II Postarchitecture, 1998 calibration.

4 Results and discusion

4.1 Results for error source (a)

Tables 2 and 3 show the relative errors before and after simulation for case (a); here the person in charge of estimation selects a different method altogether. In our case, we compare COCOMO 81 with COCOMO II.97.

Table 2: Results for error source (a) before simulation

Magnitude	**CER-1 (COCOMO 81)**	**CER-2 (COCOMO II.97)**	**Relative error** | CER1 - CER2 | CER2
Effort (person-months)	2910.18	2661	9.4%
Team Size (Number of Persons)	10	7.5	33.3%

Table 3: Results for error source (a) after simulation

Magnitude	CER-1 (COCOMO 81)	CER-2 (COCOMO II)	Relative error | CER1 - CER2 | CER2
Effort (person-months)	2352.43	2120.85	11.8%
Team Size (Number of Persons)	5	4	25%

4.2 Results for error source (b)

Similarly, Tables 4 and 5 show the results and the relative error before and after simulation for the case (b) where the person in charge of the estimation selected a different calibration to the one that he/she was supposed to select. For example, we could assume as the newest calibration as the good one; however, an oldest calibration was selected.

Table 4: Results for error source (b) before simulation

Magnitude	CAL-1 (COCOMO II.97)	CAL-2 (COCOMO II.98)	Relative error | CAL1 - CAL2 | CAL2
Effort (person-months)	3335.54	3549.22	6%
Team Size (Number of Persons)	8.83	9.21	4.12%

Table 5: Results for error source (b) after simulation

Magnitude	CAL-1 (COCOMO II.97)	CAL-2 (COCOMO II.98)	Relative error | CAL1 - CAL2 | CAL2
Effort (person-months)	2271.53	2293.38	1.00%
Team Size (Number of Persons)	4.41	4.60	4.13%

4.3 Results for error source (c)

In order to study the error source (c), we have studied what happens when two of the effort multipliers (CPLX and DATA) are selected as High instead of Very High (or vice versa). Tables 6 and 7 show the results and relative errors of the differences in selecting High or Very High as rating values for the cost drivers Complexity and Data (cf. Table 1).

Table 6: Results for error source (c) before simulation

Case	Magnitude	Sel. 1	Sel. 2	Relative error | SEL1 - SEL2 | SEL2
1	Effort (personmonths)	3614.44	4388.54	17.63%
2	Effort (personmonths)	3335.54	4116.54	19%
3	Effort (personmonths)	3549.22	4564.12	22.23%

Table 6 shows us that the major source of error comes from a wrong selection of the cost driver. As in the previous case, the error is minimized with the simulation, i.e., with the development of the system. However, the errors are still larger than in the other 2 cases, even though has attracted less attention in the literature.

Table 7: Results for error source (c) after simulation

Case	Magnitude	Sel. 1	Sel. 2	Relative error | SEL1 - SEL2 | SEL2
1	**Effort (personmonths)**	2493.15	2605.23	4.30%
2	**Effort (personmonths)**	2271.53	2347.71	3.24%
3	**Effort (personmonths)**	2293.38	2389.84	4%

In the first scenario, when COCOMO 81 effort estimations are compared with COCOMO II.97 postarchitecture, an error of 9.4% is obtained. In the next scenario, estimations provided by COCOMO II.97 post-architecture and COCOMO II.98 post-architecture are compared and a relative error of 6% in the estimations is obtained. In the third scenario, the results of the three models (COCOMO 81, COCOMO II.97 and COCOMO II.98) when estimating a project with the effort multipliers DATA and CPLX taking values *High* and *Very High* offer relative errors of 17.63%, 19% and 22.23%, respectively. The first conclusion that can be obtained is that the error introduced by choosing different ratings for the effort multipliers is higher than the error introduced in the estimations when a different model (COCOMO 81 or COCOMO II) or a different calibration (COCOMO II.97 or COCOMO II.98) is chosen. However, when the project simulator is fed with these same initial estimations, the final values obtained for the effort spent in the project have a lesser relative error than the estimates obtained with the parametric model. Hence, it can be concluded that simulation helps to minimize or reduce the effect of different sources of errors from the parametric models especially when there is some uncertainty in the rating of the cost drivers.

5 Threads to validity

Construct validity is the degree to which the variables used in the study accurately measure the concepts they purport to measure. Here, we can highlight the point that is difficult to avoid an "unfair" comparison between SD models and COCOMO, because SD models offer features that per definition are not available for COCOMO. Simulation models have many more input variables than simple estimation methods such as COCOMO, it requires more information from the analyst, but also is likely to provide results which are more certain. An interesting result when using only a COCOMO as parametric model is that the error introduced by choosing different ratings for the effort multipliers is higher than the error introduced in the estimations when a different model or a different calibration is chosen. It is to note that

Internal validity is the degree to which conclusions can be drawn about the causal effect of the independent variable on the dependent variables. A potential threat include instrumentation effect, i.e. here, we can highlight the point of how accurate is Abdel-Hamid's model.

External validity is the degree to which the results of the research can be generalised to the population under study and other research settings. A possible threat is that the results are highly dependent on the type of parametric model and of simulation model being used. The fact that simulation models are better

under uncertainty than parametric models can be supported by the type of experiments conducted. Different models have different sensitivities to their input values and this is driven in part by the variability of the data that was used to calibrate them.

6 Conclusions and future work

In this paper, an approach is shown that combines system dynamics and two versions of COCOMO to analyze the different sources of error that can occur when using a parametric estimation method. Those sources of error can be in the model used, in the calibration of the model or in the selection of the effort multipliers. To analyse these sources of error, we first generated estimations using COCOMO and evaluate how different errors or assumptions are propagated using the Abdel-Hamid's model as a way of simulating the temporal behaviour of the project. We observed how simulation helps to minimize the effect of these errors, corroborating the need for multiple estimation methods.

Future work includes to analyze the evolution of the different key variables of the software project and to determine the effect of these three sources of error during the lifecycle of the project. We will need to cover a wide variety of models and do very exhaustive sensitivity analysis to be able to reach a more solid and genearlizable conclusion.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Education project MCYT TIN 2004 06689-C03-01.

References

1. L. Farr and H. Zagorski, "Quantitative Analysis of Programming Cost Factors: A Progress Report," presented at Proceedings of the ICC Symposium on Economics of Automatic Data Processing, Amsterdam: North-Holland, Holland, 1965
2. J. Herd, J. Postak, W. Russell, and K. Stewart, "Software Cost Estimation Study - Study Results," Doty Associates, Inc, Technical Report RADC-TR-77-220, 1977
3. L. Putnam, "A General Empirical Solution to the Macro Software Sizing and Estimation Problem," IEEE Transactions on Software Engineering, vol. 4, pp. 345-361, 1978

4. B. Boehm, B. Clark, E. Horowitz, R. Madachy, R. Selby, and C. Westland, "Cost Model for Future Software Life Cycle Processes: COCOMO 2.0," in Annals of Software Engineering

   Special Volume on Software Process and Product Measurement, J. Arthur, H. S., and B. J., Eds.: AG Science Publishers, 1995

5. L. Fischman, "Calibrating a Software Evaluation Model.," presented at Proceedings of the ARMS Conference, 1997
6. D. Ferens and D. Christensen, "Calibrating Software Cost Models to department of Defense Databases - a review of ten studies," Journal of Parametrics, vol. 14, pp. 33 - 52, 1999
7. K. Mertes, D. Ferens, and D. Christensen, "An Empirical Validation of the Checkpoint Software Cost Estimation Model," Journal of Cost Analysis and Management, pp. 35 - 44, 1999
8. J. Baik, B. Boehm, and B. M. Steece, "Disaggregating and Calibrating the CASE Tool Variable in COCOMO II," Ieee Transactions on Software Engineering, vol. 28, pp. 1009-1022, 2002
9. J. J. Cuadrado-Gallego, J. Dolado, D. Rodríguez, and M.-A. Sicilia, "The second level input variables for software cost estimation models," presented at 14th International workshop on software measurement IWSM-Metrikon, Berlin (Germany), 2004
10. J. J. Cuadrado-Gallego, A. d. Amescua, J. García, and E. Ernica, "The Cost Estimator DOCU: An Empirical and Theoretical Study," presented at 15th International forum on COCOMO and software cost estimation., Los Angeles, California (USA), 2000
11. DoD, Parametric estimating handbook, 2nd Edition.: Department of Defense (DOD), United States, 1999
12. J. W. Forrester, Principles of Systems: Norwalk, CT: Productivity Press, 1961
13. B. W. Boehm, Software engineering economics. Englewood Cliffs, N.J.: Prentice-Hall, 1981
14. B. W. Boehm, Software Cost Estimation with COCOMO II. Upper Saddle River, NJ: Prentice Hall, 2000
15. R. J. Madachy, "System Dynamics Modeling of an Inspection-Based Process," presented at Software engineering, Berlin, 1996