2015 IEEE 10th International Conference on Global Software Engineering Workshops

Analogy-based software development effort estimation in global software development

Manal El Bajta
Software Project Management research team
ENSIAS, Mohammed V Souissi University
Rabat, Morocco
manal.elbajta@gmail.com

***Abstract*—*Context:* Software development has always been characterised by certain parameters. In the case of global software development, one of the important challenges for software developers is that of predicting the development effort of a software system on the basis of developer details, size, complexity, and other measures. *Objective:* The main research topics related to global software development effort estimation are the definition and empirical evaluation of a search-based approach with which to build new estimation models and the definition and empirical evaluation of all available early data. Datasets have been used as a basis to carry out an analogy-based estimation using similarity functions and measures. *Method:* Many of the problems concerning the existing effort estimation challenges can be solved by creating an analogy. This paper describes an enhanced analogy-based model for the estimation of software development effort and proposes a new approach using similarity functions and measures for software effort estimation. *Result*: A new approach for analogy-based reasoning with which to enhance the performance of cost estimation in distributed or combined software projects dealing with numerical and categorical data. The proposed method will be validated empirically using The International Software Benchmarking Standards Group dataset as a basis. *Conclusion*: The proposed estimation model could be a useful approach for early stage effort estimation on distributed projects.**

***Keywords:* global software development, effort estimation, analogy-based estimation**

I. Introduction

Software development effort estimation for GSD concerns the prediction of the effort needed to develop a global software project [1]. Development effort is considered to be one of the major components of software costs, particularly as regards global development, and it is usually the most challenging effort to predict. Several methods with which to support project managers when estimating software development effort for GSD projects have been proposed in the last few years. In particular, data-driven methods exploit historical data projects in order to estimate the effort needed during a new distributed software project under development [2]. Methods that are typically used for estimation are linear regression and case-based reasoning. These data contain cost drivers, which are information about relevant factors and the effort actually made to develop the projects. A data-driven method usually attempts to explain the relationship between the effort and the cost drivers by building an equation that models the estimation and that is used to predict the effort needed during a new project.

The main purpose of this work is to develop a software cost estimation technique for GSD projects that will allow managers to rapidly and accurately enact and assess proposed changes. In summary, this work aims to:

- Investigate how software project management approaches can be incorporated into a GSD approach.
- Develop an interactive technique with which to visualise the impact of cost drivers on GSD projects.
- Produce a tool support for the technique.

II. Background

Despite the fact that a considerable amount of research has been conducted in the context of software cost estimation for a single site, little research has been performed in the context of global software development. In practical terms, software cost estimation techniques can be classified in three main categories:

A. Expert Judgment

In this technique [3] an expert in software development processes estimates software development parameters. The accuracy of these estimations greatly depends on the degree to which a new project matches the expert's experience and the ability. The study of Jørgensen [4] shows that the expert judgement led to more accurate effort estimates. Also, a recent experiment has shown a high degree of inconsistency in expert judgment-based estimates of software development effort [5].

B. Algorithmic models

The algorithmic model [6] is designed to provide mathematical equations that can be used to perform software estimation. These mathematical equations use inputs and are based on research and historical data. The model operates on independent variables such as cost drivers. One of the well known models for estimation is the COCOMO model developed by Barry Boehm [7].

C. Non algorithmic models

Unlike the Algorithmic models, this group contains models that are based on analytical comparisons and inferences [8]. Non Algorithmic models cannot be used before first obtaining information about similar previous underestimated projects and the estimation process in these methods is usually carried out according to the analysis of previous datasets. Estimation by

978-1-4799-9874-6/15 $31.00 © 2015 IEEE
DOI 10.1109/ICGSEW.2015.19

analogy is one of the classified techniques and is basically a form of Case- Based Reasoning [9]. Estimation based on analogy is accomplished at the system and subsystem levels. By assessing the results of previous projects, we can estimate the cost and effort of a similar project. The steps in this method consist of:

- Choosing the right analogy
- Investigating similarities and differences
- Examining analogy quality
- Providing the estimation

III. Motivation

The development of software products in a cost effective manner is one of the most important goal for each organisation. The main goal is precisely the accurate estimation of the amount of effort needed to realise the projects. Many research studies have proved that without careful planning and realistic estimations the projects exceed the allocated budgets and proposed completion time [10].

Effort estimation methods basically fall into two categories: the mathematically-based category, which constructs and represents the amount of effort required on the basis of mathematics, and the experience-based category, which aims to supply the information needed to perform the effort estimation process, depending mainly on experience [11]. Experience has shown that none of the techniques can be considered to outperform any of the others. Estimation by analogy has [12], however, generated better results in the case of distributed projects.

Experienced-based effort estimation methods can be supported by the history of completed projects. For example, The International Software Benchmarking Standards Group (ISBSG) provides information for more than four thousand completed software development projects. The ISBSG contains a large deposit that helps when performing the analysis, estimation, comparisons and benchmarking of different trends in software projects (ISBSG, 2012).

ISBSG has descendent benefits that can be exemplified by a simulation approach that will help assessors to consolidate their assessment of the time required to complete a given software development project [13]. What is more, ISBSG provides three different anticipations values that represent the minimum and maximum amounts of time in which a project should be completed.

IV. Analogy-based software cost estimation model

Estimation by analogy is essentially a form of Case-Based Reasoning (CBR). For cost estimation, CBR is based on the fact that similar software projects have similar costs. In the case of a Global software project, we must first describe a set of attributes that should be relevant and independent. It is then necessary to determine the similarity between the candidate project and each project in the database. Finally, the known effort values from the historical projects are used to derive an estimate for the new project. Fig.1 shows the Framework of analogy-based estimations.

A. *Relevant attributes*

The goal of this step is to characterise all global software development projects by means of a set of attributes. Selecting attributes that accurately describe software projects is one of the most complex tasks in the analogy procedure. In our case, the aim is to estimate software cost for distributed projects. The attributes must consequently be relevant for both cost estimation and global development tasks. All relevant and independent attributes must be taken into account and these attributes must be comprehensive, implying that they must be well defined. The ISBSG dataset used to enrich our study includes a vast number of project attributes related to the application domain, programming language used, language type, development technique, resource level, functional size of the software produced, etc. The dataset contains numeric, categorical, multi-categorical and null values, and we therefore performed the following in order to better filter a large number of attributes out of the set [14]:

- We excluded attributes that were measured after project completion and would not therefore be practical for the cost estimation model that we were building.
- We calculated the number of null records for the remaining attributes and excluded those that had more than 40% null records because they would lead to a dramatic decrease in the useful sample finally used.
- We excluded records that contained null values in numerical attributes.

The attributes selected and used from The filtered ISBSG dataset, along with their description, are summarised in Table I.

Table I. Software Cost Attributes for the ISBSG Dataset

Attribute Name	Description
How Methodology Acquired	Whether the development methodology was purchased or developed in-house, or a combination of these.
Adjusted Elapsed Time	Total elapsed time for the project in calendar months.
Adjusted Function Points	The adjusted functional size of the project at the final count.
Year of Project	Derived from implementation date.
Development Type	Whether the development was a new development, enhancement or re-development.
Organization Type	The type of organization that submitted the project. (e.g.: Banking, Manufacturing, Retail).
Development Technique	Techniques used during development.
Functional Sizing Technique	The technology used to support the functional sizing process.
Recording Method	Method used to obtain effort data.
Resource Level	People whose time is included in the work effort data reported.
Max Team Size	Maximum number of people that worked on the project.
Average Team Size	Average number of people that worked on the project.

B. *Similarity function*

This step is based on the choice of a software project similarity measure. There are two popular similarity functions, namely Euclidean similarity (ES) and Manhattan similarity (MS), which describe projects by means of numerical data. These measures evaluate the overall similarity of two projects Px and Py, where “i" represents the index of cost drivers.

Euclidean distance:

- Represents a 2-norm distance
- Measures the straight-line distance

$$D(P_x,P_y)=\sqrt{\sum_{i=0}^{m}(P_{xi}-P_{yi})^2}$$

Manhattan distance:

- Represents a 1-norm distance
- Measures the absolute distance

$$D(P_x,P_y)=\sum_{i=1}^{m}\left|P_{xi}-P_{yi}\right|$$

C. *Performance evaluation*

Various modelling techniques have been applied in order to carry out software effort estimation [15]. Here, we first show the performance indicators commonly used in literature.

The estimator needs to pass through a validation set for its performance to be evaluated, and a number of performance indicators can be calculated during this connection. For each entry in the validation set, let us suppose that the estimation is fi’, while the actual effort value is fi. The magnitude of relative error (MRE) is then defined as [16]:

$$MRE_i=\frac{\left|f_i'-f_i\right|}{f_i}$$

The mean magnitude of relative error (MMRE) is defined as the mean of all the MREs produced by the evaluation set, where n is the number of data entries in the evaluation set.

$$MMRE=\frac{1}{n}\sum_{i=1}^{n}\frac{\left|f_i'-f_i\right|}{f_i}$$

Prediction level Pred is also used to test the performance of the model. It is defined as: Pred(p), which measures the percentage of estimates that fall within a range of p% and the difference on either side of the actual value. It has been suggested that a good prediction system should satisfy Pred(20) > 75% [17], but this is often difficult to achieve in software metrics estimation. Pred(30) is another commonly chosen measurement [18].

D. *Development of model*

In this part we briefly describe the framework of the estimation model. As a step towards developing the model, the high level requirements were identified as follows:

- Computation of complexity metric for input cases.
- Prediction based on analogy that uses various similarity measures.
- Updating of database as new cases are generated.
- Modification or removal of a particular record.
- Updating of mechanism used to add new similarity measures and additional parameters.

The estimation model for the proposed system is shown in Fig.1:

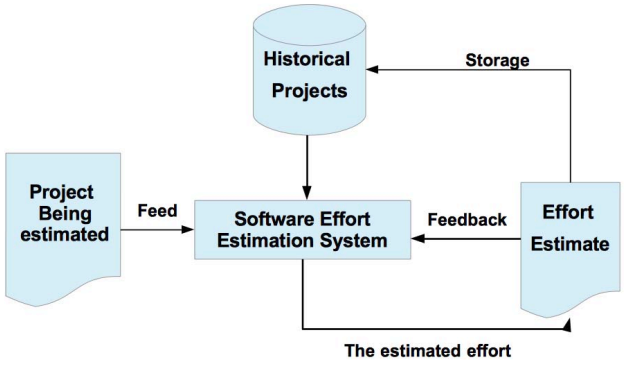

Figure 1. Framework for the Effort estimation system

Once the result has been estimated, it is added to the database to enhance the accuracy of future estimations. The objective is to accurately estimate the effort needed for a distributed project and use the results in future estimations. Increasing the volume of a database is the main objective of global software development . The larger the database, the more likely it is that the results will be accurate.

V. Conclusion

The globally distributed environment has many challenges and factors. Further research lines should therefore be followed in order to investigate the challenges and factors that impact on effort estimation approaches. Furthermore, since effort estimation methods constantly underestimate the amount of effort- time required to complete a given software development project, the authors suggest that the process of effort estimations should be improved in the context of Globally Distributed Software Development.

We have traced and presented an analogy-based model within cost drivers related to distributed projects. This approach allows the estimator to help the estimated values to

be approached when designing a global project. However, the list of inputs to the model is probably not exhaustive and will be completed as the project and our work progress. The criteria used in this study provide visibility of our scale factors and cost drivers. Other criteria may be identified during the project which will allow a new presentation of the model. Some of these criteria are the degree of consideration of risks and uncertainties.

ACKNOWLEDGEMENTS

This work has been supported by the Erasmus Mundus program Action 2: EU Mare Nostrum UE-MARE NOSTRUM (204195-EM-1-2011-1-ES-ERA MUNDUS-EMA21) and the project 'DataMining for Software Project Management' financed by the University Mohammed V of Rabat. This work is also partially supported by the GEODAS-REQ project (Spanish Ministry of Economy and Competitiveness and European Fund for Regional Development, TIN2012-37493-C03-01). The author also wishes to acknowledge the assistance of Dr. Ali Idri in developing this paper for publication.

REFERENCES

[1] C. E. L. Peixoto, J. L. N. Audy, and R. Prikladnicki, "Effort estimation in global software development projects: Preliminary results from a survey," in *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*. IEEE, 2010, pp. 123–127.

[2] A. Lamersdorf, J. Münch, A. F.-d. V. Torre, C. R. Sánchez, and D. Rombach, "Estimating the effort overhead in global software development," *arXiv preprint arXiv:1401.2730*, 2014.

[3] P. Faria and E. Miranda, "Expert judgment in software estimation during the bid phase of a project–an exploratory survey," in *Software Measurement and the 2012 Seventh International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2012 Joint Conference of the 22nd International Workshop on*. IEEE, 2012, pp. 126–131.

[4] M. Jørgensen, "A review of studies on expert estimation of software development effort," *Journal of Systems and Software*, vol. 70, no. 1, pp. 37–60, 2004.

[5] S. Grimstad and M. Jørgensen, "Preliminary study of sequence effects in judgment-based software development work-effort estimation," *IET software*, vol. 3, no. 5, pp. 435–441, 2009.

[6] B. K. Singh and A. Misra, "Software effort estimation by genetic algorithm tuned parameters of modified constructive cost model for nasa software projects," *International Journal of Computer Applications*, vol. 59, no. 9, pp. 22–26, 2012.

[7] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost models for future software life cycle processes: Cocomo 2.0," *Annals of software engineering*, vol. 1, no. 1, pp. 57–94, 1995.

[8] V. Khatibi and D. N. Jawawi, "Software cost estimation methods: A review 1," *Journal of Emerging Trends in Computing and Information Sciences*, 2011.

[9] A. Idri, A. Abran, and T. M. Khoshgoftaar, "Fuzzy case-based reasoning models for software cost estimation," in *Soft Computing in Software Engineering*. Springer, 2004, pp. 64–96.

[10] B. Boehm, C. Abts, and S. Chulani, "Software development cost estimation approachesÑa survey," *Annals of Software Engineering*, vol. 10, no. 1-4, pp. 177–205, 2000.

[11] A. Idri, A. Abran, and T. M. Khoshgoftaar, "Estimating software project effort by analogy based on linguistic values," in *Software Metrics, 2002. Proceedings. Eighth IEEE Symposium on*. IEEE, 2002, pp. 21–30.

[12] M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation," *Information and Software Technology*, vol. 54, no. 8, pp. 820–827, 2012.

[13] M. Muhairat, S. Aldaajeh, and R. E. Al-Qutaish, "The impact of global software development factors on effort estimation methods," *European Journal of Scientific Research*, vol. 46, no. 2, pp. 221–232, 2010.

[14] E. Papatheocharous, H. Papadopoulos, and A. S. Andreou, "Software effort estimation with ridge regression and evolutionary attribute selection," *arXiv preprint arXiv:1012.5754*, 2010.

[15] S. MacDonell and A. Gray, "A comparison of modeling techniques for software development effort prediction," 1998.

[16] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A simulation study of the model evaluation criterion mmre," *Software Engineering, IEEE Transactions on*, vol. 29, no. 11, pp. 985–995, 2003.

[17] A. B. Nassif, L. F. Capretz, and D. Ho, "Software effort estimation in the early stages of the software life cycle using a cascade correlation neural network model," in *Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), 2012 13th ACIS International Conference on*. IEEE, 2012, pp. 589–594.

[18] J. J. Cuadrado-Gallego, M.-Á. Sicilia, M. Garre, and D. Rodríguez, "An empirical study of process-related attributes in segmented software cost-estimation relationships," *Journal of Systems and Software*, vol. 79, no. 3, pp. 353–361, 2006.