Volume 5, Issue 2, February 2015 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Analogy Based Effort Estimation by using Use Cases and Actors

Chetan Nagar
Research Scholar (Ph.D Student)
Mewar University
Gangrar, Chittorgarh, Rajasthan, India

Dr. D. S. Rao
Department Of Computer Science and engineering
Indore Institute of Science and Technology
Indore, M.P., India

Abstract— Analogy based efforts estimation is one of the very popular method of estimation .In analogy we estimate the efforts of new project on the basis of the old project. The problem with analogy based estimation is how to select a project from the many old projects. This paper is suggesting the way of selecting a project from the old project. The selection of project depends on what project information is stored in the database. This paper is also suggesting that what information of project should be stored in the database. This paper is suggesting that projects can be compared on the basis of Use Case and Actor instead of comparison on the basis of features and functions. Almost every project using Use Case diagram in expressing the requirement and it also provides the basis for the other design diagrams. Use Case shows the functional requirement of the system. For the comparison with next projects we need to store name and description of the Use Case, the description contains the functionality in detail. One thing must be considered that Use Case should be simplified, means each use case should describe the one requirement; it should not have the relationship like generalization, include and extend. Each use case must be simple and specific. The same rule is applicable for the Actors also.

Index Terms—Use Case, Actor, Analogy, old project,Next Project.

I. INTRODUCTION

Software development efforts estimation is the process of predicting the effort required to develop or maintain software. Estimated efforts are used as input to project plans, schedule, budgets, investment analysis, pricing processes and bidding rounds. The effort invested in a software project is probably one of the most important and most analyzed variables in recent years in the process of project management.

Effort estimation methods are classified into algorithmic models and non- algorithmic models. Algorithmic models use mathematical formulae to estimate the effort and software cost. Some of the algorithmic models are COCOMO, Function Point method, Use Case Point method, Lines of Code, etc. But these methods have certain disadvantages such as they need to be adjusted according to the environment and they are not understandable. Non-algorithmic models are developed to overcome these drawbacks. Non-algorithmic models are easier to understand and are capable of exhibiting the association between cost drivers and effort. They also have the ability to extract values from databases. They are generally based on prediction process.

The success of a software development project is largely depends on the effectiveness of the effort estimation process [1].

Barry Boehm proposed a new method called COCOMO that utilized some experimental equations to estimate software development effort [6]. Introduction of the Function Point (FP) as a metric for software size estimation by Albrecht [7] was the other important event in that decade. Analogy Based Estimation (ABE) was proposed as a comparative method in 1997 [4].

Software management is an organizational and technological challenge to the project manager. During early phase of software development, it is necessary to define how to determine and allocate required resources for software projects. To achieve success in managing software development, IT companies may need to adopt a credible software cost estimation model. In practical terms, the common model of software cost estimation used in the industry is Estimation by analogy (EA). In one sense, it is formal estimation with expert judgment which can be viewed as a systematic development of the expert's opinion through experienced learning and exposure to analogue case studies [4].

It is based on underlying assumption: the more similar the software project description attributes are, the more similar the software project cost is. EA involves number of decisions that should be made by the project manager. The decisions have significant impact on accuracy of analogy estimation model including characteristics of software projects, case selection, feature selection, similarity measurement, case adaptation and adaptation rules [5]. Amongst them, similarity measurement between two software projects is the key accuracy of software prediction because it attempts to retrieve the most similar historical case to new case. But, in many circumstances the use of retrieved cases without adapting them may leads to potential overestimation or underestimation problems. The most difficult activity in analogy estimation is how to derive a new estimate from retrieved cases. Thus, it fits the case in hand and minimizes the variation between current case and retrieved case.

Use case modeling is an accepted and widespread technique to capture the business processes and requirements of a software application project. Since Use case provides the functional scope of the project, analyzing their contents provides valuable insight into the effort and size needed to design and implement a project. In general, projects with large, complicated Use cases take more effort to design and implement than small projects with less complicated Use cases. The Use Case Points (UCP) method provides the ability to estimate the man hours of a software project requires from its use cases. Based on work by Gustav Karner [8], the UCP method analyzes the use case, actors, scenarios, and various technical and environmental factors and abstracts them into an equation.

II. PROPOSED WORK

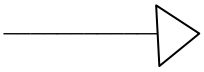
Analogy based efforts estimation is one of the very popular method of estimation .In analogy we estimate the efforts of a new project on the basis of the old project. The problem with analogy based estimation is how to select a project from the many old projects. This paper is suggesting the way of selecting a project from the old projects. The selection of project depends on what information of project is stored in the database. This paper is also suggesting that how to store the information of the project in the database. Every project has a particular type and it is known as the domain of the project. So the domain of the project must be stored in the data base as a project attribute. A company may have such type of several projects so the question is that which projects can be selected for the analogy. This paper suggests to select any 5 latest successful projects of that particular domain. Now the question is which projects are to be considered as successful projects. So the answer is that which are completed on time, within the budget and on the requirement of the customer. We are using the term latest, because latest project already gone through the new technological, business and political issues. Some of the managers were not agree on the five projects; they think three projects were sufficient for the analogy. We are also not rigid on that particular figure.

Some of the companies prefer to store the information with respect to SRS; means in terms of the requirement. So we have to compare the requirements of new project with the old selected project and find the percentage of matching. If it is matched completely then assigns the same amount of efforts and if it is not matching then find percentage of matching and predict the efforts required to build the remaining requirement. After the comparison with all the selected projects we can find that which project has higher percentage of the match. Sometimes the information is stored in terms of functions and features so the time matching is done on the basis of the features and function of the project.

This paper is suggesting a new way of storing the project information which is in terms of Actors and Use Case. Although the Use Case also represent the functional requirement but it is much simpler to understand and more objective as compare then SRS.

Use case diagram captures the system or subsystem behavior. It represents the interaction between the actors and pieces of functionality called use cases. An actor is an idealization of a role played by an external person, process, or thing interacting with the system, subsystem, or class [1], [3]. Actors participate with one or more use cases by exchanging messages. Actors may be defined in generalization hierarchies, in which an abstract actor description is shared and augmented by one or more specific actor descriptions. An actor may be a human, a computer system, or some executable process. An actor is drawn as a small stick person with the name below it. Use Case is a coherent unit of externally visible functionality provided by a classifier (called the subject) and expressed by sequences of messages exchanged by the subject and one or more actors of the system unit [2], [3]. A use case can participate in several relationships; in addition to association with actors (see Table 1).

TABLE I. RELATIONSHIP IN USE CASES

Relationship	Function	Notation
Association	To indicate the communication between actors and uses cases.	_____
Extend	To indicate the insertion of additional behavior into a base use case.	----------> << extend>>
Include	To describes a behavior that is inserted explicitly into a base use case.	----------> <<Include>>
Use case or actor generalization	To indicate the communication between a general use case (actor) and a more specific use case (actor) that inherits and adds features to it.	

Every Project uses use case diagram in expressing the requirement and in providing the basis for the other design diagrams. In analogy based estimation two things are most important, one is how to store the information and second how to retrieve the information. In concern to this paper we have to think that what information of use case should be stored in database for future reference. Use case shows the functional requirement of the system. So we have to store name and description of the Use case, the description describes that functionality in detail. One thing we must note that use case must be simplified; means each use case must describe the one requirement; it should not have the relationship like generalization, include and extend. Each use case must be simple and specified. The same rule is applicable for the actor also.

The following attributes of the project must be recorded into the database (see Table 2).

TABLE II. PROJECT ATTRIBUTE

Attribute	Description
Domain name	It shows specific type of the project
Name of the project	Every Project has a name may given by the Customer or by the developer
Project ID	It is unique by which project can identifies uniquely in the database

Domain name allow us to find a particular type of project in the database. Every project has so many Use Cases and Actors, in the following way we have to store the actors and use cases.

TABLE III. ACTOR AND USE CASE DESCRIPTION

Name of Use case/Actor	Type	Description
Login	Simple	It will Take User Id and password as input and verify it through database and if it correct then allow user to access next window.

III. WORKING OF PROPOSED MODEL

It is obvious question that how this approach will work. By one table we would like to show that how it can be work or implemented. First we need to make a detail Use Case diagram of the new project and then select one project for analogy and construct the following table.

TABLE IV. COMPARISON TABLE

Name of Use Case Of Old Project	
Description	
Efforts Spend to build that requirement	
Name of Use Case Of New Project	
Description	
If 100% matching then copy the efforts	
Else See the percentage of matching and then predict the efforts required to build the remaining portion.	
Total efforts	

Take the sum of Total efforts and we got efforts required t build the project.

The Matching of description means both the Use Cases are exactly representing the same requirement. If it is 100% matching then we can write the same efforts which way estimated for the previous project. One thing we must keep in mind that if matching is below 60or 70% then we have to predict the efforts for that use case from scratch.

Use Case has a limitation; it cannot represent the non functional requirements of the project. If project has some non functional requirement them we have to take care of it separately.

IV. RESULT

Use case diagram is the best and accurate way to express the requirement, so instead of storing the information in form of function and features, we have to store the project information in form of use case and actor.

Various researchers have used different error measurements. One of the criteria for the evaluation of cost estimation is the Mean Absolute Relative Error (MARE).

$MARE = MRE/n$

Where „n „is the number of projects in a dataset and

$$MRE = \sum_{i=1}^{i=n} \frac{(\text{act i} - \text{est i})}{\text{act i}}$$

Where est_i is the estimated effort from the model and act_i is the actual effort, and n is the number of projects in the model.

For an accurate model the MRE must be approximate to zero .Ideally it is preferred to zero but it is practically not possible. The main disadvantage of analogy based estimation is that it requires a large amount of data to compare with the next project. Whenever any new concept is suggested by the researchers, it phases the problem to test it.

This concept is recently developed so we do not have a large database to compare with next project. So to check the result of this approach first we have to make create a database in which we store the data of the project in form of Use Case and Actor, and then we can get information to compare with next project and to check that how much this approach is effective.

V. CONCLUSION

The main aim of this paper is to make analogy based estimation more accurate. Analogy based estimation is one of the simple method of estimation, but there is only one issue is that we do not know what information must be stored and which projects should be selected for analogy. This paper provides a good approach for both the issues .It suggests that information can be stored in form of use case and actor and it also suggests that how to retrieve the information for analogy.

ACKNOWLEDGEMENT

We are very much thankful to the all project managers who were involved with us in this research and they had shared their data and approach of the estimation.

REFERENCES

[1] Y. Liang, From Use Cases to Classes: a Way of Building Object Model with UML, Journal of Information and Software Technology, Vol. 45, 2003, pp. 83-93.

[2] J W. Boggs and M. Boggs, Mastering UML with Rational Rose, SYBEX Inc., 2002.

[3] J. Rumbaugh, I. Jacobson and G. Booch, The Unified Modeling Language Reference Manual, 2nd ed., Addison-Wesley, Boston, MA, USA, 2004.

[4] Shepperd, M. J., Schofield, C. 1997. Estimating Software Project Effort Using Analogies, IEEE Trans. Software Eng. 23, 736-743.

[5] Mendes, E., Mosley, N., Counsell, S. 2003. Do adaptation rules improve web effort estimation?, In Proceedings of the fourteenth ACM conference on Hypertext and hypermedia (Nottingham, UK). 173-183

[6] B. W. Boehm, "Software engineering economics," *Englewood Cliffs: NJ: Prentice Hall*, 1981.

[7] A. J. Albrecht and J. A. Gaffney, "Software function, source lines of codes, and development effort prediction: a software science validation," *IEEE Trans Software Eng. SE*, vol. 9, pp. 639-648, 1983.

[8] Karner, Gustav. "Resource Estimation for Objectory Projects." Objective Systems SF AB, 1993.

[9] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 19

