CHAPTER 5

CHALLENGES IN INTRODUCING A NEW SOFTWARE ESTIMATION TECHNIQUE

5.1 INTRODUCTION

From the vast literature survey and analysis done using questionnaire – 1, it has been concluded that it is very difficult to estimate effort for software. Software effort estimation is one of the biggest challenges that many software industries are facing because the requirements and problem vary from one project to another project. There is no standard tool for developing the software metrics model.

Software effort estimation is the process of predicting the total effort needed to develop a software project in terms of its cost, effort and schedule. The same estimation model can also be extended to estimate or predict the resources needed for the software. The research on this particular topic was started in the year 1960. In the last decade until the year 2008 researches in SCE became numerous side by side with the rapid exponential improvement in software and information technology industry.

Currently, Software effort estimation model has gained more attention from the software engineering researchers to produce a reliable model which can predict the cost of the software.
5.2 SOFTWARE COST ESTIMATION TECHNIQUES

Lodewijk Bergmans and Mehmet Aksit (2000) have dealt with many software effort estimation techniques which are available in the field of software engineering. Many companies follow different software estimation techniques depending upon their project requirements. So, because of that there is no unique software effort estimation model to suit the entire project. This has become a great challenge of developing a new tool which suits all the projects.

Software effort estimation technique is a blend of many techniques, ideas, concepts and even various models. There are many classifications of software effort estimation techniques. It can be classified into various techniques depending upon the authors. For example, Briand et al (1999) classify the software effort estimation into six categories like model based, expert, learning oriented, dynamic based, regression based and composite Bayesian.

A hybrid tool has been proposed which combines the concept of effort estimation and risk assessment techniques.

5.3 CURRENT OPEN RESEARCH IN THE FIELD OF SOFTWARE EFFORT ESTIMATION

Software effort estimation is an investigating field and many researchers are analyzing the field of estimation. The importance of software effort estimation is due to the fact that it plays an important role in the decision making process. Some of the challenges faced in developing an estimation model are given below:-
5.3.1 Metrics conversion

Since the nature of estimation process in software industry is different from the rest of science branches, it yields a lot of sizing techniques “LOC, function points, object points, application points, use case points”. Unfortunately there are no specific rules to convert between these metrics, For example, “if the estimation process for system x using function points yield 100 function points while for the same system x using use case points yields 140 use case points. Is 140 use case points equal to 100 function points?”. Answering this kind of question requires a lot of industrial data for thousands of software projects to verify from the consistency of the currently used sizing techniques, metrics and make sure that the estimation process is consistent using more than two different sizing techniques. If this issue can be solved, it will improve the correctness rate for the software estimation process and eliminate some not useful sizing techniques, Chung et al (1995). Furthermore, it increases the accuracy of the estimation process since it can be carried on in more than one metric.

5.3.2 Sizing from requirement & design phase

Software estimation applied at early stage of software development, so capturing the estimation from the known characteristics of software will be a useful approach for estimation, Demirors et al (2000). Many tools used to represent the requirement and the design of any software such as UML diagrams, Z specification language, state transition diagrams, and data flow diagrams.

Proposing a new functional sizing technique based on one of the known characteristics of the requirement & design phase will produce an efficient new sizing techniques. In order to increase the robustness of
this new functional sizing techniques, it is advised to build a sizing technique based on two or more different well-known tools; for example, propose a hybrid techniques based on the requirement and design phases such as use case and class diagram and make use of information which is available in these two diagrams, handling it in some way to compute the cost estimation of the software.

5.3.3 Complexity analysis of software systems

Complexity is a variable and subjective idiom which can be viewed in many different ways; simply, the more complicated the software yields more defects, the more difficult it will be to maintain and update. In software industry the most used formula to measure the complexity of software code is the Cyclomatic and Halsted formula.

Many sizing techniques applied in software estimation process include a lot of adjustment factors like function point, use case points, the value of these adjustments factors is assigned according to the experience of the estimator with some other defined rules. No complete specified rules or standard control the assignment process of these factors, rather these factors don’t use any of the complexity measures to assign values. So more research should concentrate on investigating the proper way to assign the adjustment factor values by highlighting a standard to control this process.

Solving the subjectivity problems for the adjustment factors and the complexity factors in the sizing techniques will lead to improve the correctness rate for those sizing techniques.
5.3.4 Risk & ROI “Return of Investment” analysis and software project cost estimation

In software development process many projects failed or overrun its budget and resource for commonly known reasons that occurred in all the development companies. So the research should take into consideration these common factors and try to include it in the estimation process with a margin of occurrences possibility to avoid overrun in estimation.

Return on Investment as a general concept is used in economics to measure the profits of any product and its revenue after launching the product in the market. Similarly in software industry the estimation process should calculate and predict the ROI as a metrics for any software before the development takes place so that management and the customer can decide upon this new suggested metrics how much it will cost as a management and as a customer and how much they will bid for this software.

Developing the suggested metric ROI-Point should measure revenue and cost, and it should go beyond that to measure the criteria which affect in an indirect way customer satisfaction, software quality maintenance, operation cost, software improvement on the organization and reliability.

5.3.5 Investigating Open Source Software – (OSS)

As the wheel of Open Source Software (OSS) technology runs, there is a need to develop a model that can estimate the effort and cost for OSS.

Investigating OSS has another face. The researches on OSS should concentrate on investigating the current available open source system by collecting the data and analyzing them to identify some interesting
pattern, then modeling them deterministically or statistically and coming up with deep insights of OSS in terms of cost estimation process.

For example in the case of the operating system Linux, it can be investigated to check the relation between the size complexity and the efforts needed to update it as long as the operating system Linux is growing and being updated, rather than the ability to check many different forms of complexity on OSS.

5.3.6 Revaluation of Function points

Function points are one of the most accepted and robust sizing techniques used in the software cost estimation process. A function point formulated by Albercht (1979), was established in the early of 1970 and since that time it has not been modified to reflect the current status of technology. These kinds of calibrations improve the accuracy of function points and show a need for a revaluation model for the function points.

Revaluation of function points needs lots of numerical data taken from large suitable projects from the entire world to reflect the international view of the current technology. Rather the model should be customized to accept any local data, and then the model automatically generates the weights for this local data.

Any evaluation of function points should specifically evaluate the 14 General System Characteristics (GSC) to check if it really reflects the current status of software development technology.

Another topic to be investigated in the evaluation of functions points is testing if all function points have same size. Some function points, for example, need 30 hours per staff to be accomplished. Another may
need 40 hours per staff. This issue should be tested to check if this variance comes from the different size of function points or if it is related to the variability in productivity rate correlation.

Function points has proven its accuracy for decades, but some research should be oriented to investigate the accuracy of function points and relate it to the programming language used in the implementation. For example, .Net IDE provides many facilities in developments which increase the productivity rate of developers unlike some other used languages. So there is a need for an adjustment factor which takes into consideration the programming language used in development of function points estimation process. This research needs a lot of historical information regarding a project developed in many languages and it should statistically build an identifier such as Backfiring index that computes function points based on LOC mathematically. This proposed identifier needs a continuous evaluation to adopt any changes in the programming languages releases.

5.3.7 Challenges in Estimation

Software effort estimation needs more efforts from the researcher either in academic or in industrial field to cooperate together to enhance the estimation process and produce the best results.

There is a need to improve the well known software cost estimation models such as COCOMO model. Most of these models require a historical data to forecast SCE, COCOMO model for example was published by Barry W. Boehm’s in 1981. Since then it has been calibrated and modified via fuzzy logic, neural network and some other techniques to adopt the changes in the software industry and project management. There is a need for a framework which can handle any changes in the calculations of these models to be up to date with the current technology.
Another urgent need in information business world today is the accurate estimates of web projects. There is a need to collect information about the web projects which developed using conceptual modeling techniques and then measure it using the functional size measurement methods for web application such as OOmFPweb to estimate project indicators such as size, effort, cost and duration Barrow et al (1993). All these models were analyzed and it was concluded to develop a tool which integrates the software effort, cost and risk.