Journal of Computer Science 7 (11): 1679-1684, 2011
ISSN 1549-3636
© 2011 Science Publications

An Analysis of Hybrid Tool Estimator: An Integration of Risk with Software Estimation

[1]J. Frank Vijay and [2]C. Manoharan
[1]Department of IT, Panimalar Engineering College, Chennai
[2]Annai Mathammal Sheela Engineering College, Namakkal Dt, Tamil Nadu, India

Abstract: Problem statement: One important problem with software development projects is to get an early and nevertheless accurate estimation of the effort needed to complete the project within the schedule. In the literature various methods have been developed for this purpose. The most popular examples are Boehm's COCOMO, Albrecht's Function Point Method or Sneed's object-point method. The two last named methods are based on early results of an analysis phase; whereas COCOMO is based on an a priori estimation of the software size in "Lines of Code". Despite of the increasing needs and available tools and methods, a satisfactory solution is still to be found. During the last two years, has gained some interest in this community an approach based on hybrid technique of software estimation. **Approach:** In this study, we discuss that traditional Function Point Method does not cover the quality factors and the estimation is fully based on development of systems. Hence, the quality assurance factors were discussed in this study. The comparative analysis of the existing software estimations were also developed and compared with the developed model so that the efficiency of the model can be analyzed with the existing methods. The classification of software system for which the effort estimation is to be calculated is based on the COCOMO model classes. So, our aim is to develop a hybrid method which combines all the important parameters from the various existing method for effort estimation. Once the effort estimation has been found, the same have been extended to risk assessment techniques by considering various risk parameters. So, the developed hybrid model is an integrated model of estimation with risk assessment. **Results:** A software has been designed (Front End-Java, Back End-MS-ACCESS) which shows the comparison between the traditional Function point method and the proposed method. **Conclusion:** Detailed comparative analyses have been made based on the result for all the estimation techniques.

Key words: COCOMO model classes, hybrid technique, effort estimation, quality parameters, estimation techniques, traditional function, hybrid method, comparative analysis

INTRODUCTION

There are two main dimensions in project metrics. The first dimension main concentrates on the development schedule so that we can avoid delay and potential problems and risks. The second dimension main focuses on the quality of the software so that the developed software is of good quality and it satisfies the customer. So, most of the organizations focuses on the above said two dimensions. There are many effort estimation techniques for software systems developments are available (Vijay and Manoharan, 2009; 2010). But none of the models paid attention to schedule and quality parameters. So, we have developed a new hybrid estimation technique which is fully focussed on assuring the quality in effort estimation for software system development. In this study, we have created a new hybrid model which estimates the effort, schedule and quality parameters. The entire results of the developed hybrid model have been illustrated in the results section. An example software system used to apply the proposed study is the testing software. In the results comparisons section, the effort (in terms of person-months) is used to compare the various results of some available models with the proposed result. In the conclusion section, the results between proposed and existing scenario are compared with the detailed graphs and performance chart.

MATERIALS AND METHODS

Function points: Function-oriented software metrics is used to measure the functionality delivered by the application as a normalization value. The most widely

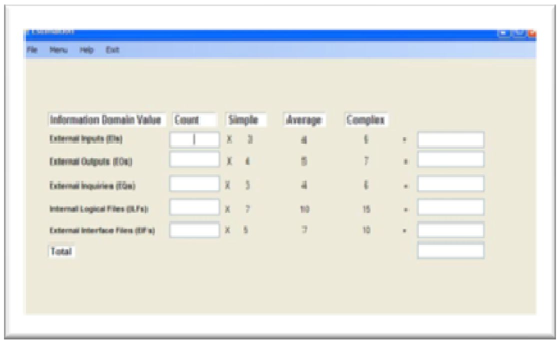
Corresponding Author: J. Frank Vijay, Department of IT, Panimalar Engineering College, Chennai, Tamil Nadu, India

used function-oriented metric is Function Point (FP). The traditional Functional Point metric method does not take into account the quality parameters of the software and moreover the values that been assigned to the parameters will be decided by the project managers experience. So, the values for the parameters will vary depending upon the manager's interest on the project. As a net result, it is very difficult to estimate the software by these varying parameters. So, we can't use this value for estimating, the schedule, effort needed, project time, cost estimation etc. So, we have made a vast literature survey by means of questionnaire to various software companies. The questionnaires were given only to the project managers who are familiar in their projects. From their ideas and suggestions, we have derived the parameter values and by using those values the effort will be calculated automatically. This effort estimated value also covers some of the important quality parameter values. So, we have also made a comparative analysis of the existing functional point model with the developed model which covers the quality aspects of the software. The detailed output of the derived model is shown below in the form of snapshots.

Analysis of the model: In this derived hybrid model, the effort estimation for particular software has been calculated by using the following methods:

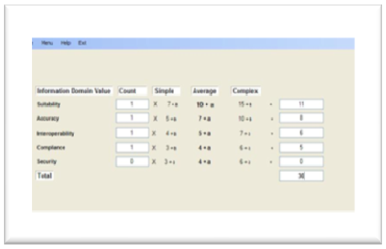
- Automated Hybrid Model (New Model) (Vijay and Manoharan, 2009)
- Use Case Point Method Model (Developed to suit for English Statements) (Keung *et al*., 2004)
- COCOMO Model.
- Function point model (Manual-existing model)
- Revised Functional point model (New model)
- Lines of code

Software estimation has been calculated (Fig. 1-10) be using all these models.



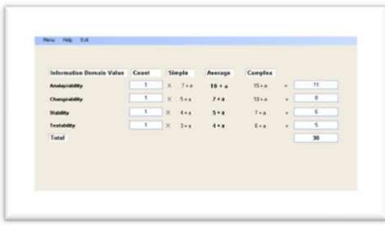
Information Domain Value	Count		Simple	Average	Complex		
External Inputs (EIs)		X	3	4	6	=	
External Outputs (EOs)		X	4	5	7	=	
External Inquiries (EQs)		X	3	4	6	=	
Internal Logical Files (ILFs)		X	7	10	15	=	
External Interface Files (EIFs)		X	5	7	10	=	
Total							

Fig. 1: Parameters estimation chart



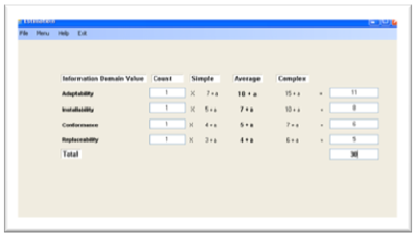
Information Domain Value	Count		Simple	Average	Complex		
Suitability	1	X	7+a	10+a	15+a	=	11
Accuracy	1	X	5+a	7+a	10+a	=	8
Interoperability	1	X	4+a	5+a	7+a	=	6
Compliance	1	X	3+a	4+a	6+a	=	5
Security	0	X	3+a	4+a	6+a	=	0
Total							30

Fig. 2: Parameters calculation chart



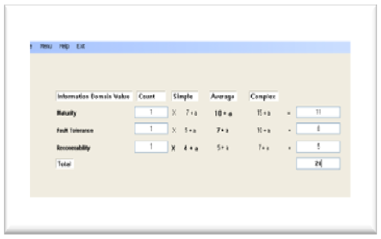
Information Domain Value	Count		Simple	Average	Complex		
Analyzability	1	X	7+a	10+a	15+a	=	11
Changeability	1	X	5+a	7+a	10+a	=	8
Stability	1	X	4+a	5+a	7+a	=	6
Testability	1	X	3+a	4+a	6+a	=	5
Total							30

Fig. 3: Total weight calculation chart



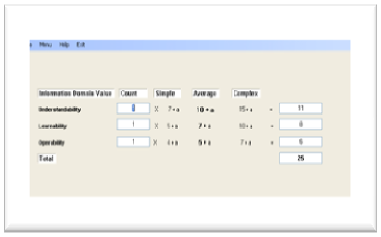
Information Domain Value	Count		Simple	Average	Complex		
Adaptability	1	X	7+a	10+a	15+a	=	11
Installability	1	X	5+a	7+a	10+a	=	8
Conformance	1	X	4+a	5+a	7+a	=	6
Replaceability	1	X	3+a	4+a	6+a	=	5
Total							30

Fig. 4: Parameters calculation chart



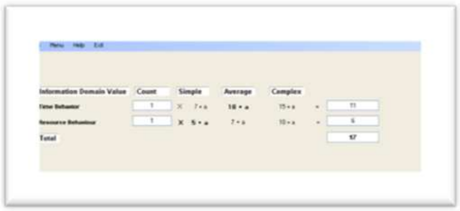
Information Domain Value	Count		Simple	Average	Complex		
Maturity	1	X	7+a	10+a	15+a	=	11
Fault Tolerance	1	X	5+a	7+a	10+a	=	8
Recoverability	1	X	4+a	5+a	7+a	=	5
Total							24

Fig. 5: Total weight calculation chart



Information Domain Value	Count		Simple	Average	Complex		
Understandability	1	X	7+a	10+a	15+a	=	11
Learnability	1	X	5+a	7+a	10+a	=	8
Operability	1	X	4+a	5+a	7+a	=	6
Total							25

Fig. 6:Total weight calculation chart



Information Domain Value	Count		Simple	Average	Complex		
Time Behavior	1	X	7 • a	18 • a	15 • a	=	11
Resource Behaviour	1	X	5 • a	7 • a	10 • a	=	6
Total							17

Fig. 7: Parameters estimation chart

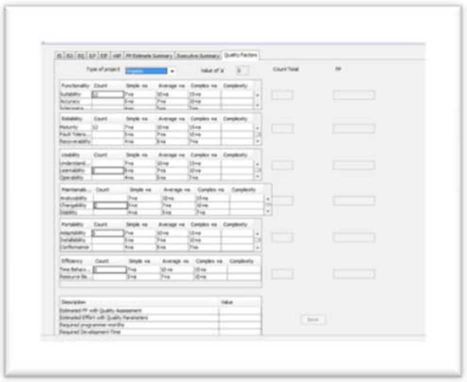


Fig. 8: Total estimation calculation values

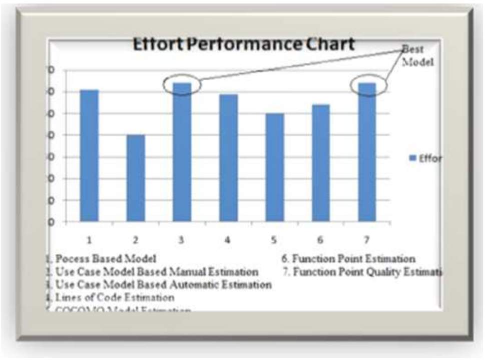


Fig. 9: Effort comparison chart

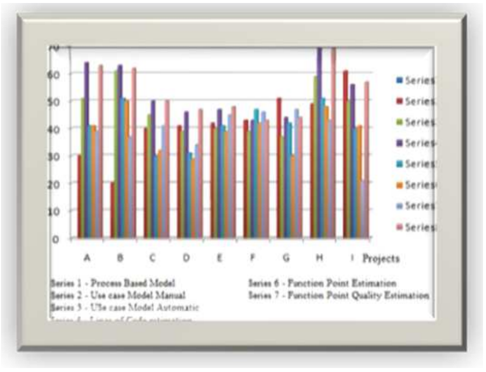


Fig. 10: Software tools estimation comparison chart

By using the results that we got by using the above said model, we have made a comparative study in the form of a graph. The output of the result is given below.

Integration of software risk with estimation: Once the effort has been found by using one of the above said estimation methods. The User can select the method needed for estimating the software manually using menu-driven option from the tool. Once the effort has been found, then the result has to be integrated with risk.

Software risk assessment: Large software projects have a very high frequency of schedule overruns, cost overruns, quality problems, and outright cancellations. Although software cost estimating is a very difficult intellectual problem, there are three fundamental equations that are linked together for estimation. They are given below:

Size of deliverable/assignment scope = staff
Size of deliverable/production rate = effort
Effort/staff = schedule

Functions of hybrid tool: Step 1: Sizing specifications, source code, and test cases: The first step in any software estimate is to predict the sizes of the deliverables that must be constructed. Sizing must include all deliverable such as specifications, documents, and test cases as well as source code. As of 2008, sizing is a standard feature of commercial software cost estimating tools, and a variety of sizing methods are now included, such as:

- Sizing based on function point metrics
- Sizing based on Lines Of Code (LOC) metrics

It should be noted that one very common risk with estimates based on "lines of code" metrics is that such estimates are not reliable for predicting user documents or any non-coding activity such as quality assurance, data base administration, and project management. LOC-based estimates and function point-based estimates are of approximately equal accuracy for predicting coding activities but the LOC estimates usually are less accurate for non-code activities. Since studystudy in all of its forms is often the most expensive task for large defence applications, this problem is fairly significant.

Step 2: Estimating defects and defect removal efficiency levels: A key aspect of software cost estimating is predicting the time and effort that will be needed for design reviews, code inspections, and all forms of testing. In order to estimate defect removal costs and schedules, it is necessary to know about how many defects are likely to be encountered. Poor quality control is another major risk that can lead to litigation. Lack of early defect detection and removal via inspections can lead to huge delays in testing schedules. What happens is that testing might start on time, but due to the unexpected volume of defects it cannot end on time. Testing is the primary phase where schedules begin to go out of control. The defect removal efficiency of each step will also be estimated. The effort and costs for preparation, execution, and defect repairs associated with each removal activity will also be estimated.

Step 3: Selecting project activities: Once the size of various deliverables has been approximated the next step is to determine which specific activities will be carried out for the project being estimated. This is one of the major areas where software cost estimating tools excel. Activity-based cost estimates with perhaps 20-25 activities are the level of precision offered by modern cost estimating tools.

Step 4: Estimating staffing levels:_ Although staffing, effort, costs, and schedules are all important for the final estimate, the normal place to start estimating is with staffing levels. The fundamental equation for determining staff is:

Size of deliverable/assignment scope = staff
The UCP tool applies this fundamental staffing equation in a wide variety of forms, including but not limited to:

Pages of specifications / assignment scope = analysts
Lines of source code/assignment scope = programmers
Test cases/assignment scope = testers
Pages of user manuals/assignment scope = technical writers
Number of employees / assignment scope = managers

Step 5: Estimating software effort: The term “effort” defines the amount of human study associated with a project. The amount of effort can be expressed in any desired metric, such as study hours, study days, study weeks, study months, or study years.
The general algorithm for predicting effort is:

Size of deliverable / production rate = staff effort
Here too this basic equation is used in a variety of forms including but not limited to:

Pages of specifications / production rate = analyst months
Lines of source code / production rate = programmer months
Test cases/production rate = testing months
Defects found/production rate = restudy months
Pages of user manuals/production rate = writing months

Step 6: Estimating software costs: The fundamental equation for estimating the cost of a software activity is simple in concept, but very tricky in real life:

Effort * (salary + burden) = cost

Step 7: Estimating software schedules: The fundamental equation for estimating the schedule of any given software development activity is:

Effort / staff = time period

Step 8: Estimating requirements changes during development: One important aspect of estimating is dealing with the rate at which requirements “creep” and hence makes projects grow larger during development. Fortunately, function point metrics allow direct measurement of the rate at which this phenomenon occurs, since both the original requirements and changed requirements will have function point counts.

Step 9: Software risk analysis: The software industry has long been troubled by major schedule slippage, major cost overruns, and a high incidence of outright failure. Of all the troublesome factors associated with software, schedule slips stand out as being the most frequent source of litigation between out sources vendors and their clients. Schedule slips are also the main reason for executive frustration with software for internal projects.

RESULTS AND DISCUSSION

The existing results for the Tested application in the software industry are given below:

- Based on the automated hybrid model (Fig. 11) estimate the effort as 63 persons per month
- Based on the use case point model, the estimated effort is 64 persons per month
- Based on the COCOMO model estimate, the estimated effort is 49 persons per month
- Based on the Function Point metric estimate, the estimated effort is 62 persons per month
- Based on the derived Function point metric, the estimated effort is 64 persons per month
- Based on the LOC estimate, the estimated effort is 61 persons per month

The average estimate (using all six approaches) is 63 person-months.

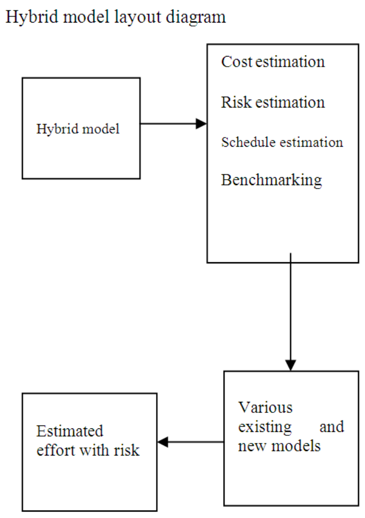

Hybrid model layout diagram
Hybrid model
Cost estimation
Risk estimation
Schedule estimation
Benchmarking
Various existing and new models
Estimated effort with risk

Fig. 11: Hybrid model layout diagram

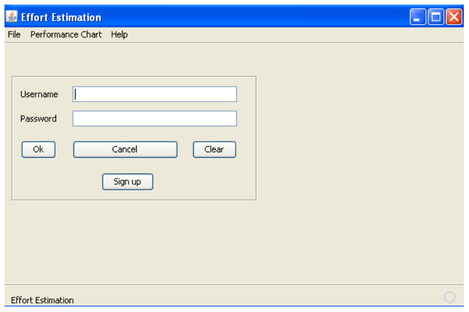
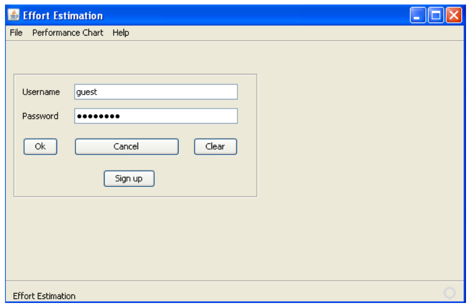


Fig. 12: Front end screen



Username: guest
Password: ••••••••

Fig. 13: Login screen

Training of model: The Hybrid model has been trained with a minimal set of keywords and parameters for a specific project. But, the model can be trained with more set of dataset so that it can suit a variety of projects.

Sample screenshots of hybrid model: The Sample Screenshots for the developed hybrid model is shown below from Fig. 12-21.

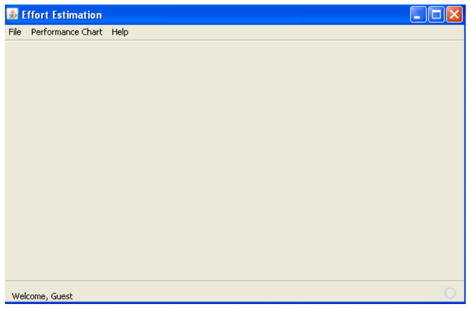
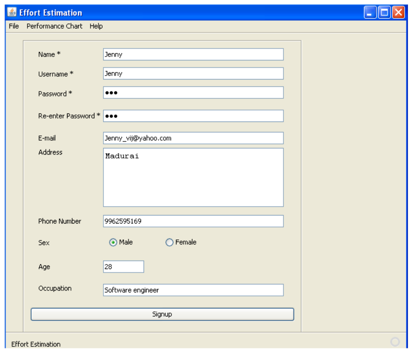
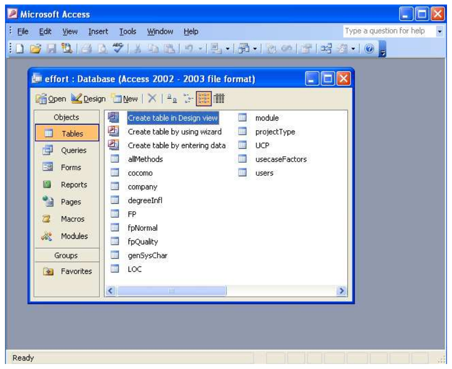


Fig. 14: Effort estimation software screen



Name *	Jenny
Username *	Jenny
Password *	•••
Re-enter Password *	•••
E-mail	Jenny_vij@yahoo.com
Address	Madurai
Phone Number	9962595169
Sex	[✓ Male] Female
Age	28
Occupation	Software engineer

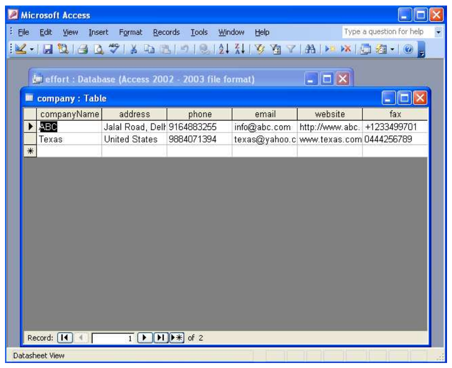
Fig. 15: Signup details



effort : Database (Access 2002 - 2003 file format)

Create table in Design view
Create table by using wizard
Create table by entering data
allMethods
cocomo
company
degreeInfl
FP
fpNormal
fpQuality
genSysChar
LOC
module
projectType
UCP
usecaseFactors
users

Fig. 16: Database file formats

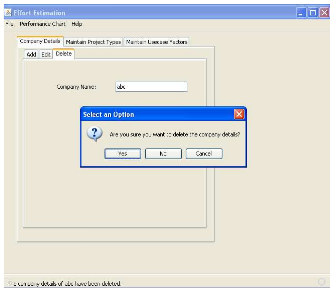


effort : Database (Access 2002 - 2003 file format)

company : Table

companyName	address	phone	email	website	fax
ABC	Jalal Road, Delh	9164883255	info@abc.com	http://www.abc.	+1233499701
Texas	United States	9884071394	texas@yahoo.c	www.texas.com	0444256789

Fig. 17: Database attribute screen



Company Name: abc

Are you sure you want to delete the company details?

The company details of abc have been deleted.

Fig. 18: Company details form

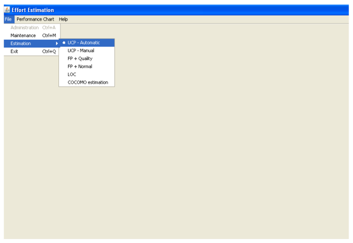
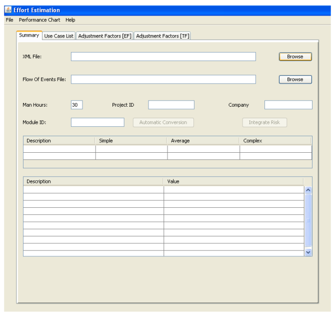


Fig. 19: Software estimation tools menu



XML File:

Flow Of Events File:

Man Hours: 30

Project ID

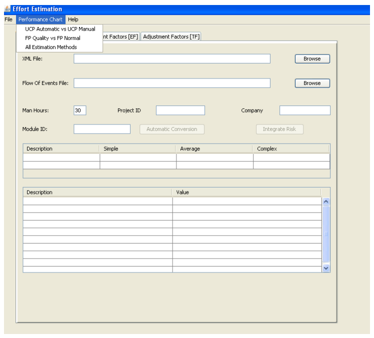
Company

Module ID:

Description	Simple	Average	Complex

Description	Value

Fig. 20: Automated hybrid tool menu



XML File:

Flow Of Events File:

Man Hours: 30

Project ID

Company

Module ID:

Description	Simple	Average	Complex

Description	Value

Fig. 21: Parameters estimation table

Advantages of hybrid model: The main advantages of this developed hybrid model are given below:

- This model uses six different cost estimation techniques
- This model integrates the software estimation with the risk assessment strategy
- This model gives a detailed explanation of the obtained output with the relevant graphical explanation
- The user can select the estimation technique which he/she is interested
- The calculated effort from this hybrid method, has been extended to various formulas for analyzing the risk strategy of the software
- This model can be used for decision making purposes

CONCLUSION

Based on the above results, the proposed 64 person-months of effort is nearer value to the average result of other estimation models. And hence this type of estimation may be recommended for the software development. The unique difference between the proposed and existing estimation of effort for the software system development is the level of quality consideration. That is, existing estimations are using only few quality factors for effort estimation, but the proposed effort estimation covers the important quality factors, which automatically reflects in the development of software. Other metrics may be used to estimate the effort and substituting other quality factors can be explored as a future scope.

REFERENCES

Keung, J., R. Jeffery and B. Kitchenham, 2004. The challenge of introducing a new software cost estimation technology into a small software organisation. Proceedings of the Software Engineering Conference (SEC'04), IEEE Xplore Press, USA., pp: 52-59. DOI: 10.1109/ASWEC.2004.1290457

Vijay, J.F. and C. Manoharan, 2009. Initial hybrid method for analyzing software estimation, benchmarking and risk assessment using design of software. J. Comput. Sci., 5: 717-724. DOI: 10.3844/jcssp.2009.717.724

Vijay, J.F. Frank, V.J. and C. Manoharan, 2010. A comparative analysis of software engineering with knowledge engineering. J. Comput. Sci., 6: 1208-1211. DOI: 10.3844/jcssp.2010.1208.1211