Research Journal of Applied Sciences 2 (2): 214-218, 2007

A Tool for Software Effort Estimation

T.S.V. Mani
Sri Krishna College of Engineering and Technology, Kuniamuthur,
Coimbatore-641 008 Tami lNadu, India

Abstract: Exactness in software cost and effort estimation is tedious task in software development. Too many variables-human, technical, environmental and political-can affect the ultimate cost of software and effort applied to develop it. However, software project estimation can be transformed from a black art to a series of systematic steps that provide estimates with acceptable risk. Four module assignment factors, *team size, concurrency, intensity* and *fragmentation* were identified as potentially significant determinants of software development effort. This system is the generalized system, which can be used in any type of software development industry. The system has been developed for software development industry to reduce the scheduling effort for developing software. The results of the study indicate the work assignment factors (team size and concurrency) can be used to improve the predictive ability of effort estimation.

Key words: Concurrency, team size, activity network, concurrency, intersity, fragmentation

INTRODUCTION

In software development organizations, there is seldom a one to one mapping between software developers and development tasks. It is frequently necessary to concurrently assign individuals to multiple tasks and to assign more than one individual to work cooperatively on a single task. A principle goal in making such assignments should be to minimize the effort required to complete each task. Developers are assigned to tasks on the minor assignment factors, team size, concurrency, intensity and fragmentation. Since team size and concurrency are major two factors, these are considered for investigation. This research generates an activity network for any software project. The network generated is divided into four parts namely analysis part, design part, coding part and implementation part. In each part the number of professionals are specified, so that they will work concurrently in that part and the minimum time is calculated. This process will be repeated by varying the professionals and minimum time is calculated. From these solutions (say 10 or 15), the solution, which is having minimum time is said to be the optimal solution.

MATERIALS AND METHODS

Literature survey: In most development organization, there is seldom a one-to-one mapping between the software developers and development tasks. It is frequently necessary to concurrently assign individuals to multiple tasks and to assign more than one individual work too cooperatively on a single task. Depending on the collection of tasks and developers, it may be possible to assign tasks to different developers in a number of different ways. As an extreme example, the entire team of developers may be assigned to work on each task, proceeding on the next task only after completely finishing of the previous one. At the other extreme, it might be also possible to rank the various tasks in terms of complexity and assign more complex tasks to subsets of the more experienced developers.

It is a well developed axiom in project management that increases the number of people working concurrently on a task does not result in a corresponding increase in productivity (e.g., Brook's Law: Adding more programmers to a late project makes it later, do not consider the configuration of task assigned as a factor in predicting software development effort. The results of an empirical study exploring the impact of four task assignment factors, team size, concurrency, intensity and fragmentation, on software development effort to the intermediate COCOMO model, it estimates are improved significantly for the study project.

FACTORS OF EFFORT ESTIMATION

A key underlying concept in all following definitions is the notion of a time unit, which is a discrete interval of time of some fixed duration. A task is characterized by its duration, the span between the time unit, when work on a task is begun and the time unit, when the task is completed. Our effort estimation factors are defined with respect to tasks (i.e., they are properties of tasks, rather than of individuals). Four module assignment factors,

team size, concurrency, intensity and fragmentation were identified as potentially significant determinants of software development effort.

Team size factor addresses the cumulative number of programmers working on a single module over all active time units for that module. Each programmer reporting work on that module during at least one of its active time units is counted once. Thus, everyone reporting any activity at all for a particular module is counted as a member of the development team for that module. These alone are insufficient to determine the expected effect of TEAM on development effort.

Intensity factor measures the degree of schedule compression. Intensity can be defined as the ratio of the number of active time units to the total number of time units in the development span.

Concurrency factor is an integral part of managing the flow of work between team members, previously found critical to team performance. Specifically concurrency may be viewed as the degree to which team members work together or independently on module. It is derived by examining individual time units and determining the number of programmers working simultaneously during those time units.

Fragmentation factor examines the degree to which a team's time is fragmented for a module over multiple modules.

STATEMENT OF THE PROBLEM

Generation of activity network: There are four stages, namely

- Requirements analysis
- Design analysis
- Coding
- Testing/Implementation

For these four stages an activity network is drawn. From the network, minimum time and cost are found for each stage. The network is drawn based on the predecessor/successor Table 1.

Team size: If the team size exceeds/decrease, the project completing time may reduce/increase and cost is also increase or decrease. This work will give varieties of project completing time and project cost.

Concurrency: This project is divided into four phases as described above. These phases will be applicable for all standard programs from which the effort is estimated. In each phases concurrency process is involved. For example in MRP, there will be various departments in which materials will be required. Here each department requirements will be treated as an activity in the network generation.

Table 1: Difference between PERT network and this activity network

Pert network	Activity network
Only one way is possible	More than one way is possible
No feed back	Feed back is available
Only one stage is available	More than one stage is available

Our whole network will be divided into four phases, in each phase we find the minimal time and minimal cost and that time and cost will be send as input to another phase and from there minimal time and cost will be found. This process will be repeated for testing and implementation phase, at last we obtain the optimal solution.

This may be repeated with varying team size and overall optimal time for completing the project is found.

Objectives of the study:

- To use the factors (that often involves in software development) team size and concurrency at a greater extent.
- To enhance the efficiency of software development industry in delivering the project to its customer at the right time.
- It focuses on efficient scheduling process, to fasten the project management.
- Decrease in time and costs for meetings, conferences during the software development.

Selection of the system: This system is the generalized system, which can be used in any type of software development industry. Scheduling the project in the industry is a difficult task and also selecting the software professional to the right task is little bit difficult work. Moreover delivering the project at the right time to the customer is important, this may delay, in some software industry, due to the scheduling the project, so this tool will be very useful for them. So to overcome these difficulties, this tool may be implemented in the software development industry. OOPS language is selected for its platform independent feature.

RESULTS AND DISCUSSION

A screen is designed for getting the values for pred and succ for generating the activity network as shown below.

Pred	Succ
1	2
1	3
1	4
2	5
2	6
2	7
3	8
3	9
3	10
4	11
4	12
4	13
5	14
6	14
7	14
8	15
9	15
10	15
11	16
12	16
13	16
14	17
15	17
16	17
17	18

Another input screen is designed for getting the cost for each activity as shown below

Pred	Succ	Cost
1	2	5
1	3	3
1	4	2
2	5	6
2	6	3
2	7	4
3	8	7
3	9	5
3	10	6
4	11	4
4	12	3
4	13	2
5	14	7
6	14	5
7	14	3
8	15	2
9	15	4
10	15	6
11	16	7
12	16	8
13	16	9
14	17	10
15	17	12
16	17	15
17	18	10

Another screen is designed for getting the number of persons available for each stage. Activity network is generated as an output for the given nodes and list of the total cost as an output is also displayed. A sample of outputs for the four given inputs is shown below and the optimal solution is also shown.

Sample input 1

Enter number of analyst: 1
Enter number of designers:1
Enter number of software engineers: 1
Enter number of persons in implementations: 1
The total cost for the above parameters:

Analyst	Designers	Developers	Testing	Totalcost
1 (50)	1 (51)	1 (37)	1 (10)	148

Sample input 2

Enter number of analyst: 3
Enter number of designers: 5
Enter number of software engineers: 2
Enter number of persons in implementations: 1
The total cost for the above parameters:

Analyst	Designers	Developers	Testing	Totalcost
3 (22)	5 (16)	2 (22)	1 (10)	70

Sample input 3

Enter number of analyst: 2
Enter number of designers:3
Enter number of software engineers: 3
Enter number of persons in implementations: 1
The total cost for the above parameters:

Analyst	Designers	Developers	Testing	Totalcost
2 (28)	3 (29)	3 (15)	1 (10)	82

Sample input 4

Enter number of analyst: 2
Enter number of designers:4
Enter number of software engineers: 2
Enter number of persons in implementations: 1
The total cost for the above parameters:

Analyst	Designers	Developers	Testing	Totalcost
2 (28)	4 (24)	2 (22)	1 (10)	84

Optimal solution

The optimal solution to develop the project is

Analysts	: 2
Designer	: 3
Developer	: 3
Tester	: 1

The above persons may be involved to develop the project !!!

PROGRAM SPECIFICATION

Activity network: Scheduling of a software project does not differ greatly from scheduling of any multitask engineering effort. Therefore, generalized project scheduling tools and techniques can be applied to software with little modification.

Program Evaluation and review technique (pert) and critical path method (cpm) are two project scheduling methods that can be applied to the software development. these methods provide quantitative tools that allow the software project to Hale *et al*. (2001) determine the critical path-the chain of tasks that determines the duration of the project; (Hassan, 1999) establish most likely time estimates for individual tasks by applying statistical models and (Hong and Danfeng) calculate boundary times that define a time window for a particular tasks.

Figure 1Activity networks are graphical notations, which are used to illustrate the project schedule. Activity networks show the dependencies between the different activities making up the project. For example, consider

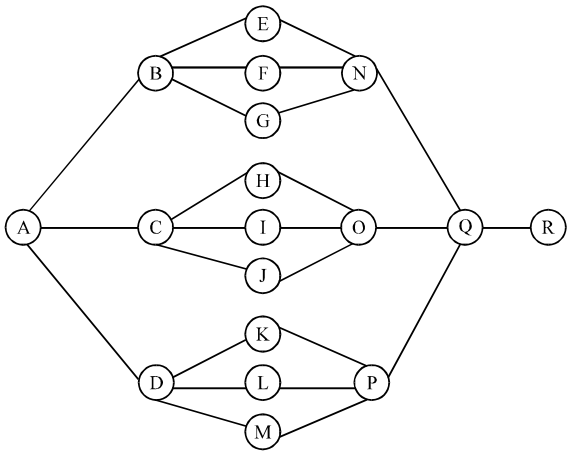

Fig. 1: Activity network

the set of three activities T1, T2, T3 with costs 8, 15, 15, respectively and also T3 depends on T1. In this case, after completion of T1, T3 will be activated.

Pert charts are a more sophisticated for of activity chart in which, instead of making a single estimate for each task, pessimistic, likely and optimistic estimates are made. There are therefore many potential critical paths. These depend on the permutation of estimates for each activity. Critical path analysis in pert charts is therefore very complex.

The network generated for this project is almost same as the pert network but in our network more than one way to traverse the network is possible and feedback is available, i.e., we can move from one activity to another activity in one direction and also in the reverse direction also, in order to calculate the total cost of the network. in our network more than one stage is possible.

The model: The network generated for this study is divided into four stages, namely, requirements analysis, design analysis, coding, testing/implementation. The activities are given as input in the form of nodes as predecessor and successor. An activity connects the predecessor and successor nodes. These four stages or phases are very important in any software development. The first part of our network constitutes the requirements analysis, followed by the design analysis. The third part of the network, followed by design analysis will be coding (the single activity from the design nodes), followed by a single activity called as the implementation stage.

The costs in the form of duration (time taken to complete the activity) are also given as input for each activity. Each stage calculates its own total cost (i.e., the time taken to complete the particular stage). These four costs are then added to produce the total time duration for completing the particular project.

The number of persons available in the particular software development industry is given as input, so that each person can be scheduled concurrently. For example, consider the analysis stage, there are three activity networks with costs 5, 3, 3, respectively. The number of analysts available is two, then these two persons are allowed to complete the first two activities with cost 5 and 3. The second person completes the second activity in three days and he will be allowed to do the third activity. Therefore, by concurrency, the total time duration for this analysis phase is 11. Like this, for all the stages the total cost is found. These total cost are then added and total cost of the network is found.

Like this, the number of persons available in the industry is varied say for example, 10 or 15 times and total cost is found for all the inputs. from these total cost, the system finds the minimum cost and suggests that the particular combination of the professionals is more suitable for developing the project.

CONCLUSION

The system has been developed for software development industry to reduce the scheduling effort for developing software. The results of the study indicate the work assignment factors (team size and concurrency) can be used to improve the predictive ability of effort estimation. Use of these effort estimation factors provided an improvement in the effort scheduling in the software development. Development effort was found to decrease with:

- Breaking work assignments down into tasks that can be accomplished individually.
- Compressing the development schedule of modules.
- Allowing teams to focus on a small number of tasks,
- Invoking concurrency in each stage.
- Varying team size in each stage.

SUGGESTIONS FOR FUTURE ENHANCEMENT

This system concentrates only in two module assignment factors such as team size and concurrency. The other two module assignment factors intensity and fragmentation is not considered in this study. In the future this system may be enhanced by considering the other two module assignment factors for studying how these factors will help in the effort estimation of software. I conclude with a ray of hope and delight that I have set a small path for other to take on from me.

ACKNOWLEDGMENT

With immense pleasure, I express my sincere gratitude to Professor S. Palaniappan, M.Sc., M. Phil., Principal, Erode Arts College, Erode, for having given his kind consent to work on this project.

I would like to express my sincere thanks to Dr. M. Arthanari, Head of the Department, Department of Computer Science, Erode Arts College, Erode, for having constant enthusiastic encouragement.

I would like to express my sincere thanks to my Mr. S. Pannir selvam, M.Sc., M.Phil., Selection Grade Lecturer in Computer Science, Erode Arts College, Erode, for having extended help, suggestions, valuable guidance, constant enthusiastic encouragement and advice throughout the project work.

I wish to express my thanks to the Librarian, Central Library, ITT, Chennai, permitting me to refer journals.

I also wish to express my sincere thanks to Mr. A.V. Ramani, H.O.D, Department of Computer Science, Sri Ramakrishna Mission Vidyalaya College of Arts and Science, Coimbatore-20, for his valuable guidance.

I wish to thank, Management trustee and Principal of Sree Narayana Guru College, for allowing me to do M.Phil degree.

I also wish to thank, Management trustee and Principal of Sri Krishna College of Engineering and technology, for allowing me to continue M.Phil. degree.

Last, but not least, I also acknowledge the help of all others who were involved in this project in one way or the other.

REFERENCES

Hale Joanne, Parrish Allen and Smith K. Randy, 2001. An Empirical Study Using Task Assignment Patterns to improve the accuracy of Software Effort Estimation. IEEE Trans. Software Eng., 27: 264-271.

Hassan Gomaa, 1999. Software Design Methods for Concurrent and Real Time Systems, Addison-Wesley Publishing Company.

Hong and Danfeng, Software Cost Estimation, Department of Computer Science, University of Calgary, Alberta, Canada.