© 2020 | International Journal of Scientific Research in Science and Technology
IJSRST | Volume 7 | Issue 3 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
DOI : https://doi.org/10.32628/IJSRST2075334

Analysis of Software Project Management Estimation Method using Function Point

Lakshmi Shanker Singh[1], Dr. Rupak Sharma[2]

[1]Research Scholar, Department of Computer Science, Monad University, N.H. 9, Delhi Hapur Road, Hapur, Uttar Pradesh, India

[2]Assistant Professor, S.R.M. Institute of Science, S.R.M. University, NCR Campus, Delhi Meerut Road, Modinagar Ghaziabad, Uttar Pradesh, India

ABSTRACT

Effort estimation has been used for planning and monitoring project resources. As software grew in size and complexity, it is very difficult to predict the development cost. There is no single technique, which is best for all situations. A careful comparison of the results of several approaches is necessary to produce realistic estimates. The use of workforce is measured as effort and defined as total time taken by development team members to perform a given task. It is usually expressed in units such as man-day, man-month, and man-year, which is a basis for estimating other values relevant for software projects, like cost or total time required to produce a software product.

Keywords : Software Project Management, Monitoring Project Resources, Software Product, Development Team

I. INTRODUCTION

The estimation of development effort is the key factor to manage and control of a software project management. The major cost factor in software industry is cost of manpower in most cases. The estimation of effort can be used for planning and monitoring project resources. In software industry, the critical issue is the accurate prediction of the software development costs. Similarly the accurate determination of how much effort and time a project required is also important for project managers, system analysts and developers. Without a reasonably accurate cost estimation capability, it will be difficult for the project managers to determine how much time and man power is required for a project.

1)

2)

1.2 SOFTWARE PROJECT MANAGEMENT

Software project management is the art and science of planning and leading software projects. Analysis of software project management shows that the following are the most common causes of failures:

1. Insufficient end-user involvement
2. Poor communication between customers, developers, users and project managers
3. Unrealistic or articulated project goals
4. Inaccurate estimates of needed resources
5. Badly defined or incomplete system requirements and specifications

6. Poor reporting of the project's status
7. Poorly managed risks
8. Use of immature technology
9. Inability to handle the project's complexity

Most of the software is produced to meet the client's requirements. Due to the frequent changes and progresses of the technology in the software industry, the experience of one product may not be applied to the other one. These are the main risk in the software development process hence it is crucial to manage software projects efficiently.

Fig 1.1 Constraints for the software Projects

This figure 1.1shows the main constraints for the software projects. It is very vital for an organization to supply quality product, keeping the cost in client's budget constraints and delivers the project as per schedule. Several internal and external factors will be there to affect this constrain triangle. Any one of these factors can severely impact the others.

Software project management is needed to incorporate user requirements together with budget and time constraints. It comprises of different activities, such as planning of project, scope of software product, cost estimation, scheduling of tasks and events, as well as resource management.

Project management include the activities such as,

- Project Planning
- Scope Management
- Project Estimation

1.2.1 Project Planning

Software project planning is the phase, which is performed before the starting of actual software development process. It doesn't involve any concrete activity that has direct connection with software construction; but it is a set of multiple processes that facilitates software production.

1.2.2 Scope Management

It defines the scope of project; which incorporates all the activities concerned about a deliverable software product. Scope management is important because it generates boundaries of the project by clearly specifying what should be done in the project and what should not be done. This form the project to contain limited and measurable tasks that can easily be documented and consecutively avoids cost and time overrun.

1.2.3 Project Estimation

Accurate estimation of various measures is a must for an effective management. With faithful estimation, managers can direct and control the project more efficiently and effectively.

1.3 SOFTWARE ESTIMATION

Software estimation is the prediction of the most realistic value for different attributes/matrix of a software development such as size, effort, time of development etc. [45].

1.3.1 Software size estimation

Size of the software can be estimated in terms of Line of Code as KLOC or the number of function points in the software or the number of use case

points in the use case diagram of the software. KLOC depends upon the languages used for coding and function points may vary in accordance with the user or software requirement, Use case point may vary according to the use case diagram of the software.

1.3.2 Effort estimation

The software effort is estimated in terms of person-hour required to produce the software. For estimating the effort, the software size is needed. This software size is derived either by managers' experience or by the historical data of organization. This software size can be converted into efforts by using the prescribed formulae.

1.3.2 Time estimation

The time of development of the projects can be estimated from the size and efforts of software. The different tasks of the software are divided into smaller tasks, activities or events by Work Breakthrough Structure (WBS)[45]. The tasks are scheduled on day-to-day basis or in calendar months.

The total of time required to complete all these tasks in hours or days is the total time needed to complete the project.

1.3.3 Cost estimation

This is the most difficult task of all because it depends on more components than any of the previous ones.

The following components are required for estimating project cost [45].

- Size of software
- Software quality
- Hardware
- Additional software or tools, licenses etc.
- Skilled personnel with task-specific skills
- Travel involved
- Communication
- Training and support

3) 1.4 TYPES OF ESTIMATION MODELS

Different methods exist for effort estimation. All of them have their own advantages and disadvantages that depend on the information the effort estimator has, his experience and his judgement. The main classifications are:

- Expert estimation
- Formal estimation
- Combination based estimation

4) Expert estimation

The quantification step is based on judgmental processes. An expert on the subject of effort gives judgement on this. The different approaches under this category are:

- Web based estimation
- Group estimation

Formal estimation

Here it is based on mechanical processes, such as the use of a formula derived from historical data. The different approaches under this category are:

- Analogy based estimation
- Parametric model
- Size based estimation models
    - Function point analysis
    - Use case point analysis

Combination based estimation

The step is based on a combination of judgmental and mechanical estimates from different sources such as expert judgment based on estimates from a parametric model and group estimation

II. FUNCTION POINT ANALYSIS

Function point analysis is a method for estimating size of the software from the user's point of view. According to Roberto Meli, Luca Santillo[37] FP (Function Points) is the most widespread functional type metrics which is suitable for quantifying a software application. This metrics is based on a 5 user identifiable logical "functions". These functions can be divided into 2 data function types and 3 transactional function types.

In this approach effort can be estimated as follows:

- Estimate the number of external inputs, external interface files, external outputs, external queries and logical internal tables (files).
- Use the Function Point Conversion Factor table to get total initial function points.
- Initial function points are adjusted via 14 complexity factors to obtain final (adjusted) function points.
- Use adjusted function points to obtain KSLOC.
- Use KSLOC to estimate efforts

2.1 METHODOLOGY

There are different methods to estimate the development effort. Estimation of a software project development mainly implies the values of effort (in Person hours), time taken for development, and cost of development. For each software project, an estimation method should be selected so that it should comply with the actual values of these attributes.

2.2 PROBLEM STATEMENT

Software Estimation is an important factor in the software development life cycle, it help managers tender on projects and allocates resources efficiently. So many methods exist to estimate the software efforts. Function point analysis and use case point analysis are two important methods among them. We can identify the approximation of the methods to the actual from the comparison of these methods.

Our work tries to answer the following questions:

- How much the software development projects deviate from the original plan, with respect to cost, schedule and functionality?
- Which methods are used to estimate software effort, and whether these systematically differ in accuracy?
- How important is accurate effort estimation perceived to be, and how much is the level of accuracy considered a problem in the software industry?
- What are the main causes for software projects to deviate from their original plan?

2.3 OBJECTIVE

Comparison of function point method with use case point method of software effort estimation in order to make the decision process easy by performing case study on ten applications.

It requires the following steps:

- Study the software effort estimation
- Study the function point method and use case point method of effort estimation
- Select 10 applications for our work.
- Calculate the attributes of each application in both the method

The comparison of these attributes among both the methods of all applications

III. LIBRARY MANAGEMENT SYSTEM

This system gives the details of the books available in various departments, the transactions of books and about the book holders. It enables a fully automated library service, which makes the transactions easier and systematic.

3.1 Use case diagram

The use case diagram of library management application is shown in the Fig 3.1 and the complexities of different use cases present in the system are in the Table 3.1

TABLE 3.1. COMPLEXITY OF USE CASES: LIBRARY MANAGEMENT SYSTEMS

Sr. No	Use cases	No. of transactions	Complexity of use case
1	Registration	3	Simple
2	Enquiry	5	Average
3	Verification of Member	1	Simple
4	Issue of Book	>7	Complex
5	Availability of Books	4	Average
6	Return Book	6	Average
7	Fine Calculation	4	Average
8	Maintenance of Books	5	Average

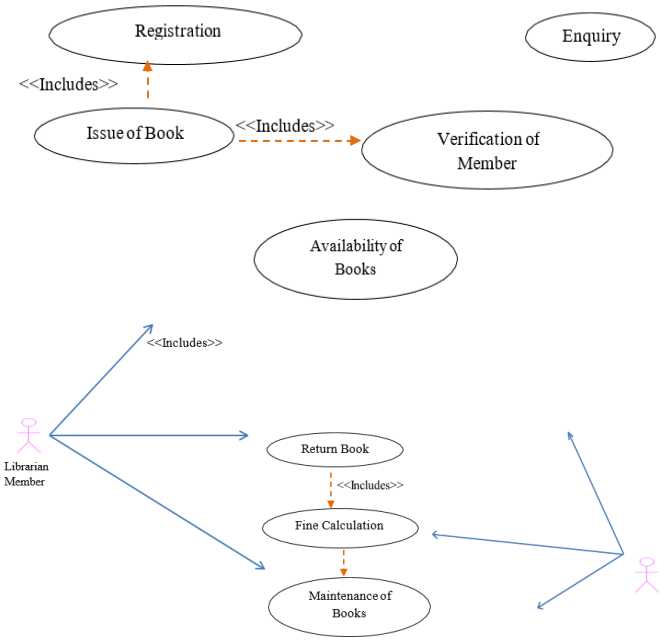

Fig 3.1: Use case diagram of Library Management Systems

Function point analysis

Formula for counting function point is

FP=UFC * [0.65+ (0.01* SFi)] Eqn (3.1)

The general characteristics [5] of the application library management system is shown in the Table

The complexity of different components of library management system and its unadjusted function count is shown in the Table 3.2

TABLE 3.2. UNADJUSTED FUNCTION COUNT: LIBRARY MANAGEMENT SYSTEMS

Type of components	No of Components	Complexity of components	Value
External inputs	3	3	1
	2	4	7
External outputs	1	4	1
	2	5	4

External inquiries	1	3	1
	2	4	1
Internal logical files	2	7	14
External interface files	1	5	5
	Total (UFC)		61

The function point count of Library management system can be calculated using the Eqn (4.1). FP = UFC * [0.65+ (0.01*SFi)]

= 61*[0.65+ (0.01*36)]
= 58.58

Function point count for Library Management System= 58.58

The effort is calculated based on Jones first order estimate [23]

It gives an estimate for optimal schedule months from the function point count. According to C. Jones [23] the appropriate exponent, j, should be chosen according to the type of system and the general capability of the development team as per Table 3.3.

TABLE 3.3 EXPONENTS FOR JONES'S FIRST-ORDER ESTIMATE[23]

Kind of software	Best in class	Average	Worst in class
Systems	0.43	0.45	0.48
Business	0.41	0.43	0.46
Shrink-wrap	0.39	0.42	0.45

The following formula converts function points into total man-months.

Effort = f^{3^j} / 27

Where, f -> function point j -> Jones first order estimate So, effort =$58.58^{3^{*0.43}}$ / 27 = 7.06 person-months

IV. CONCLUSION AND FUTURE WORK

A number of different models and effort estimation methods have been developed in the past four decades. This clearly indicates the awareness among the researchers to improve effort estimation in software engineering. There are many factors have impact on the software development process. These factors can be human, technical and their impact can never be fully predicted. We have studied the different estimation techniques and illustrated two approaches for measuring the size in the estimation process in this paper. If the estimation is done accurately, it results in error decrease. The Estimation process reflects the reality of project's progress. It avoids cost/budget or schedule overruns. This process is quite simple which takes a few inputs. This assessment framework helps inexperienced team to improve project tracking and estimation. The effort estimation still remains unreliable. Too many techniques are developed including use case point, story point etc. to overcome this inability. In the future work, we can compare these techniques for their ability.

V. REFERENCES

[1]. Boehm, B. Software Engineering Economics. Englewood CI@, NJ Prentice-Hall (198 1).

[2]. Kemerer, C.F. An empirical validation of software cost estimation models. Communications of the ACM vol. 30, no. 5 (May 1987) 416-429.

[3]. Albrecht & Gafney. Software function, source lines of code, and development effort prediction: a software science validation, IEEE Transactions on Software Engineering 9, 6 (1983) 639-648.

[4]. Mukhopadyay, T., Vicinanza, S.S., Prietula, M.J. Examining the feasibility of a case-based reasoning model for software effort estimation. MIS Quarterly (June 1992) 155-171.

[5]. Srinivasan, K., Fisher, D. Machine learning approaches to estimating software development effort. IEEE Transactions on Software Engineering, VOI. 21, no. 2 (February 1995) 126-137.

[6]. Briand, L.C., Basili, V.R., Thomas, W.M. A pattern recognition approach for software engineering data analysis. IEEE Transactions on Software Engineering, vol. 18, no. II (1992) 931-942

[7]. Allan J. Albrecht May 1984. AD/M Productivity Measurement and Estimation Validation, IBM Corporate Information Systems. IBM Corp.

[8]. Barry W. Boehm, Bradford dark, Ellis Horowitz, Chris Westland, Ray Madachy and Richard Selby. Cost Models for Future Software Lifecycle Processes: COCOMO 2.0 Annals of Software Engineering. Volume 1, pp., 57-94, 1995.

[9]. Magne Jorgensen. May 2007 Forecasting of Software Development Work Effort: Evidence on Expert Judgment and Formal Model.

[10]. Parvinder S. Sandhu, Porush Bassi, and Amanpreet Singh Brar. 2008. Software Effort Estimation Using Soft Computing Techniques. World Academy of Science, Engineering and Technology 46 2008.

Cite this article as :

Lakshmi Shanker Singh, Dr. Rupak Sharma, "Analysis of Software Project Management Estimation Method using Function Point ", International Journal of Scientific Research in Science and Technology (IJSRST), Online ISSN : 2395-602X, Print ISSN : 2395-6011, Volume 7 Issue 3, pp. 472-478, May-June 2020. Available at doi : https://doi.org/10.32628/IJSRST2075334

Journal URL : https://ijsrst.com/IJSRST2075334