Kinshuk Batabyal
Sudip Chattopadhyay
Sujoy Samaddar
Shirshendu Roy

Project Management Office(PMO)
NRI Financial Technology,
Globsyn Crystals, Tower I, 6th Floor, Block EP, Sector V, Salt Lake City Electronics Complex, Kolkata 700 091, West Bengal, India
shirshendur@nrifintech.com

DEVELOPMENT OF FRAMEWORK BASED EFFORT ESTIMETION BENCHMARK

Abstract: *In this paper, an attempt was made to estimate the effort for framework based software development in an Indian scenario. Estimations benchmark was developed for two different framework using past data and found to be effective for estimation for new development works. In this paper, only one scenario is discussed.*
Keywords: *Estimation, Frame work, Effort, Cost, FP, LOC, FPM,UCP,RACI*

1. INTRODUCTION

Managing software projects in an effective way is crucial for any software project. The success of the software project depends on the planned and hassle free execution of the project. The main goal of the software project management is to enable a group of software engineer to work effectively towards the completion of the project.

The main objective of software project management is to complete the project within the budgeted effort and schedule. In order to complete the project within effort and schedule proper estimation of the size of the work is essential. Hence, estimation accuracy is one of the important success criteria of project success.

In order to accurately estimate the size of the project and effort required for completion, some definite approach was required to develop some metric or size in terms of which we can express the size of the project and the effort required for completion of the project.

The traditional method for estimating software size count are Lines of Code (LOC) and Function Point (FP).The usages of these techniques in the projects have their own advantages and disadvantages. Lines of Code (LOC) is the simplest method of calculation of the size count but it can vary widely with the individual coding style, it do not consider the complexity and effort required while development, in case of component reuse the size will shows as low which is not correct and it is very difficult to accurately estimate LOC in the final product form the problem specification. Function Point (FP) is not dependent on the construction methods and is also independent of the language, toll and the methodologies used for the implementation. But calculation of FP is complex and varies from different classes of the systems. Feature Point Metric (FPM) it is thx tension of the FP to include new class. The feature point is the weighted sum of the algorithm and the FP [1, 2]. Another extension of the FP is Full Function Point (FFP), which is generally used for measuring the real time application. Use-Case Points (UCP) method was developed by Gustav Karner of Objectory. Use-case Points are counted from the use-case analysis of system. UCP are counted during early phases of an object-oriented project that captures its scope with use cases. Each use-case is scaled as easy, medium, or hard to produce the point count. The use-case points can be also adjusted for project's technical and personnel attributes, and directly converted to the hours in order to obtain a rough idea of a nominal project schedule [3]. While feature point and FFP extend function point, the object point measures the size from a different dimension. This measurement is based on the number and complexity of following objects i.e. screens, reports and 3GL components. This is relatively new measurement and not been yet very popular. But as it is easy to use at the early phase of the development cycle and also measures software size reasonably well so this measurement has been used in major estimation models such as COCOMO II for cost

estimation.

Software organizations are using various cost estimation models, techniques or methods [4] to estimate the exact and accurate cost that will be used in the development of software project. In case of Expert Judgment Technique, experts usually use their experience and understanding of a new project and available information about new and past projects to derive an estimate. This is dependent on the 'expert'. Another technique namely 'Delphi approach' tries to overcome some of the shortcoming of the Expert judgment approach. The coordinator prepares a summary of the responses from the experts on a form requesting another iteration of the experts' estimates and the rationale for the estimates. Bottom-up Estimating Method estimates cost of each software components and then combines the results to arrive at an estimated cost of overall project. It is more stable because estimation errors in the various components have a chance to balance out. But it is more time-consuming. Top-down Estimating Method is opposite of bottom-up method. An overall cost estimation for project is derived from global properties of software project. Then project is partitioned into various low-level components. It is faster, easier to implement but not identify low-level problems. Software Life Cycle Management (SLIM) Method, a constraint model was developed by Putnam is applied to projects exceeding 70,000 lines of code. This model is very sensitive to development time as decreasing the development time can greatly increase the person months needed for development. The problem with this model is that it is based on knowing or being able to estimate accurately the size in lines of code of the software to be developed. There is great uncertainty in the software size. So it may result in inaccuracy of cost estimation. Constructive Cost Model [5, 6, 7] (COCOMO I) is one of the best-known and best-documented software effort estimation methods. It is the set of three modeling levels: Basic, Intermediate, and Detailed. It has been experiencing difficulties in estimating the cost of software developed to new life cycle processes and capabilities including rapid-development process model, reuse-driven approaches and object-oriented approaches. For these reasons the newest version COCOMOII was developed [8].

It is obvious form the above discussion that there are different models available for estimation of the size and cost of the software development. We need to customize the model(s) or integrate available model as per our need.

2. PROBLEM STATEMENT

The organization was facing a challenge in estimating the effort required for the framework based business application development. The project teams were engaged in estimating the effort based on their experience and expertise [9]. As all these activities were happening in heuristic way the variation in estimation accuracy was very high. To make the estimation process more robust for the organization, it was decided to have homogenous benchmark data for different business application development activities for available frameworks.

This paper will discuss about the development of estimation benchmark for one of the framework say A.

2.1 Basic assumptions

Based on the meeting with the project teams (which uses the frameworks), effort data were collected for each attribute for both category of resources (experience & novice), based on following assumptions;

1. All efforts are given in Man Days (MD) only.
2. During estimation, Requirement gathering is considered 'out of scope'
3. Estimated effort covers following activities:
   i Use Case Document preparation
   ii Coding
   iii Unit Test Case Preparation
   iv Unit testing (Manual & Junit)
   v QA Test Case preparation
   vi QA Test Execution
4. Experience and Novice resource categories were defined based on the discussion with project teams.
5. Efforts of an Experienced and Novice are collected after discussion with the project teams.

2.1 Resource Categorization

As almost all project teams are working with mixed resources (i.e. novice and experience), resource categorization is key

success factor for estimation bench marking. Based on the experience level, capability of the resource may also vary. Some special and generic technological factors were taken into consideration while categorizing the resources.
In order to work in the framework based business application development projects the resources need to have some mandatory skills, summarized below;

- Basic working knowledge on framework A and Camel is mandatory
- Prior knowledge on Jasper report and Spring framework (MVC, Core, Security) is required

The criteria for resource classification is as follows.

Novice: Total software development experience < 2 years (It covers DB, Java and Java related Technology) and application development experience using framework A < 6 months.

Experienced: Total software development experience > 2 years (It covers DB, Java and Java related Technology) and application development experience using framework A > 6 months.

Based on the collected data of different activities by experienced and novice ratio analysis of required effort was done and it was observed that median ratio of effort (novice/experience) is 1.5. Hence, as per the analysis novice requires 1.5 times effort for doing similar activity as compared with the experienced professional.

3.DEVELOPMENT OF BENCHMARK

In this Section, procedure for the development of the estimation benchmark is described.

3.1 Approach

The activities which will be done for achieving the goal can be summarized as follows;

- Identify the feature list of the frameworks based on discussion with project managers (responsible for development of the framework)
- Development of data collection template
- Data collection and validation with the PMs
- Development of benchmark data
- Effort data collection form the existing projects which uses frameworks
- Comparative analysis of actual and benchmark data
- Review the benchmark data and finalization of guideline

3.2 Roles and Responsibility

The roles and responsibility matrix was develop to make sure all the stake holders are aware of their respective ownership. The RACI Matrix is given in the Table 1.

Table 1. *RACI Matrix*

RACI - Matrix						
Process steps		**PIC**				**Comments**
No.	**Description of Activity**	**PMO Team**	**PM (Developed the framework)**	**PM (using the framework)**	**Senior Management**	
1	Identify feature list of framework with project manager.	R	A	I	I	
2	Prepare the Data Collection Template.	A	C	I	I	
3	Collect actual project data (effort spent) from project team	R	A	I	I	
4	Collect actual project data (effort spent) from different project team	R	C	A	I	
5	Validate project data with project managers.	A	R	R	I	
6	Review project data and deduce the estimation benchmark	A	C	C	I	

7	Review benchmark with senior management.	R	C	I	A	
8	Discuss with project manager (using the framework) and finalize practical case studies from Project JIRA.	R	C	A	I	
9	Validate case study with benchmark data deduced.	A	C	R	I	
10	Monitor the benchmark data and finalize the guideline.	A	C	R	I	

RACI - Definition:
R = Responsible (Those who do the work to achieve the task)
A = Accountable (Those ultimately answerable for the correct and through completion of the deliverable, and the one who delegates the work to those responsible)
C = Consulted (Those whose opinions are sought)
I = Informed (Those who are kept up-to-date on progress)

3.3 Development of Formula

It was planned to create a benchmark for the different activities required for the business application development using the framework. Based on the planning data was collected form five different projects (P_1, P_2, P_3, P_4, P_5) which are using framework **A** for application development. As parallel work is being done for development of framework **A** itself, hence data form framework **A** development activity was also taken into account. Based on the expert decision, five weighted average formula were developed (mentioned in Table No- 2) for derivation of values for an attribute in the estimation template.
The finalized rule for populating the parameter value in the estimation template can be summarized as follow:

- If a project do not have estimate in any section, it will be left blank.
- It can be updated if data is available in future for the particular activity

"Weighted Average" method was followed for data normalization of different parameters. During normalization process, following rules were followed to make the normalization data more homogeneous:

- 10% effort from P_1 (i.e. framework development project for framework **A**), as this team spend time to framework development activity,
- 30% effort from P_2 & P_4, as these projects uses the most of the functionalities of framework A for application development
- 10% effort from P_3 project as this is using few functionalities framework A
- 20% effort from P_5, as most of the required functionalities are developed using framework A but the number of required functionalities are very less.

Table 2. *Weighted Average Formula*

P_1	P_2	P_3	P_4	P_5	Formula used to find estimated time
Value Present	Value Present	Value Present	Value Present	Value Present	(P_1 x 10% + P_2 x 30% + P_3 x 10 % + P_4 x 30% + P_5 x 20%)/100
Value Present	Value Present	Value Present	Value Present	Null	(P_1 x 10% + P_2 x 30% + P_3 x 10 % + P_4 x 30%)/80
Value Present	Value Present	Value Present	Null	Null	(P_1 x 10% + P_2 x 30% + P_3 x 10 %)/50
Value Present	Value Present	Null	Null	Null	(P_1 x 10% + P_2 x 30%)/40
Value Present	Null	Null	Null	Null	(P_1 x 10%)/10

3.4 Development of Benchmark

Data was collected for different parameters (as identified in the data collection template) form the different projects (as mentioned above) for different business application development. Then with the help of the weighted average formula listed in the Table No-2, the benchmark data for different activities were derived. These data are "Benchmark" data and are supposed to be used as reference point during estimation

4. VALIDATION OF BENCHMARK DATA

After development of estimation benchmark data it is necessary to validate the benchmark data. Unless the validation is done properly, it can't be said that the developed benchmark data can be successfully used as reference point for the new estimation processes.

In order to validate, data was collected for different projects which are uses framework **A** for business application development. At first estimation was done for new developments taking benchmark data as the reference point, these data was termed as the **Estimate Data**. Then actual efforts required for completion of the work were noted and were termed as **Actual Data.** Then variations were calculated for the Estimated and Actual data. It was found that the variation was within the range of ±15%, with the median value below 10%, which is not bad at the startup.

5. CONCLUSION

In this paper, an estimation benchmark procedure based on the development framework is proposed. This is considered to be an important step in new estimation reference for a particular benchmark. Based on the consultation with the project managers, senior management and team members the benchmarks for different activities have been done. Hence, it is supposed to be robust and practical. At the start up work have been done with considerable less number of data points and it is expected that with the time more data points will be available and the benchmark will be fine-tuned further. It is not a onetime activity, regular monitoring and fine tuning is required for making the benchmark current with the time.

REFERENCES:

[1] Mall, R. Fundamentals of Software Engineering, pp. 38-53.

[2] Aggarwal, K.K., & Yogesh, S. (2001). *Software Engineering.*

[3] Stępień, B. (2003). Software Development Cost Estimation Methods and Research Trends. *Computer Science*, *5*, 68-82.

[4] Merlo-Schett, N. Seminar on Software Cost Estimation WS 2002/2003, Department of Computer Science, pp. 3-19.

[5] Gao, X., Lo, B. (1995). *An Integrated Software Cost Model Based on COCOMO and Function Point Approaches.* Southern Cross University.

[6] Lee, T., Donoh, C., Baik, J. (2010). Empirical Study on Enhancing the Accuracy of Software Cost Estimation Model for Defense Software Development Project Applications ISBN 978-89-5519-146-2, 2010 ICACT, pp.1117-1119.

[7] Khatibi, V., Dayang, N.A., Jawawi, P. Software Cost Estimation Methods: A Review Vol. 2, no. 1, ISSN 2079-8407, pp. 22-24.

[8] Wu, L. The Comparison of the Software Cost Estimating Methods University of Calgary, pp. 2-12.

[9] Lionel, C.B., & Wieczorek, I. (2001). *Resource Estimation in Software Engineering*. In the second edition of the Encyclopedia of Software Engineering, pp. 1-52.