Journal of Theoretical and Applied Information Technology
15th February 2019. Vol.97. No 3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DATA ANALYSIS IN SOFTWARE EFFORT ESTIMATION USING IMPROVED DELPHI METHOD

[1]ASHISH RAI,, [2]G. P. GUPTA, [3]PANKAJ KUMAR

[1]Assistant Professor, Department of Computer Science,
Shia P.G. College, University of Lucknow, Uttar Pradesh, India
[2]Professor, Department of Computer Science,
Shia P.G. College, University of Lucknow, Uttar Pradesh, India
[3]Associate Professor, Department of Computer Science & Engineering,
Shri Ramswaroop College of Engineering and Management, Uttar Pradesh, India

E-mail: [1]arai1975535@gmail.com, [2]prof.gpgupta@gmail.com, [3]pk79jan@gmail.com

ABSTRACT

Software Estimation Accuracy is one of the most difficult tasks for the software developers. Defining the project duration, effort estimation and estimated cost, early in the development phase is greatest challenge has to be achieved for software projects. Inaccurate effort estimation of software development is one of the most important reasons of computer and IT major project failures.

Low effort estimates may lead to project management problems, delayed deliveries, budget overruns and poor software quality, too high effort estimates may lead to loss of business opportunities and improper and inefficient use of resources. The projects main focus at Simulate Research Laboratory is to improve judgment-based effort estimation methods, which is most frequently used by the software industries. By introducing better mental steps in effort estimation, we can achieve significant improvement in software development estimation processes.

There are great challenge while studying expert judgment like Delphi Estimation. To understand the use of multidisciplinary competencies, especially financial resources enables studies in realistic software development process, psychology and software engineering.

This paper explores the relationship between development effort, team size and software size. The main objective of this research is to improve the existing Delphi method for the estimation of software development effort using hybrid approach. Proposed, improved method has been validated by using 15 NASA project dataset and the results show that the improved Delphi method for software effort estimation resulted in slightly better as compared to results obtained earlier.

Keywords: *Effort Estimation, Productivity, Algorithmic Model, Variance, MMRE, Pred.*

1. INTRODUCTION

Proper analysis and Effort Estimation is necessary for successfully planning for a testing project. Any flaw in critical estimation phase, results into the missing of the project deadlines, and also reduces Return of Investment and loses of customer's faith. However in my view "Bad estimation can lead to poor distribution of work".

Software metric and especially software estimation is based on measuring of software attributes which are typically related to the product, process and the resources of software development. This kind of measuring can be used a parameters in project management models which provide assessments to software project managers in managing the software projects to avoid problems such as cost overrun and delay in schedule. Underestimating the costs may result in management approving proposed systems

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

which can exceed their budgets, with underdeveloped functions and poor quality, and failure to maintain the time factor. Overestimation may result in too many resources committed to the project, or, during contract bidding, result in not winning the contract, and also lead to loss of jobs. Accurate and reliable effort estimation is still one of the most challenging processes in software engineering. There have been number of attempts to develop cost estimation models.

Most of the traditional techniques, such as function points, regression models, COCOMO etc. require a long term estimation process. Effort and schedule overruns are serious problems in the software industry. In the most popular software textbooks and also representative set of software estimation research papers, the systematic shortcomings in use of estimation terminology have been found.

The Delphi Method is Looping process which is used to collect and filter the judgments of experts by using a series of questionnaires with feedback. The questionnaires are prepared to discuss the problems, opportunities, solutions or predictions. Each questionnaire is prepared on the output and the results of the previous questionnaire. This process continued for number of times until the research question is appropriately answered. The Delphi Method can be used whenever there is incomplete knowledge about a problem or phenomena to focus the intelligentsia or expertise on the problems in hand. Formal software development effort estimation models have been around for more than 40 years. In spite of massive effort and promotion, available formal estimation models are not in much use, so it is time to focus industrial estimation process improvement work and scientific research on Judgment-based effort estimation methods. There are very good reasons to claim that future estimation process improvement and research initiatives should aim at better judgment-based effort estimation processes and not at better formal models. The relation between effort and size in software development contexts is not stable.

2 DELPHI METHOD

2.1 Existing Delphi Method

The Delphi method has been used in research to develop, identify, forecast and to validate in a wide variety of research areas. Three round Delphi is typical, single and double round Delphi studies have also been completed. The number of experts, vary from 4 to 17. The method can be modified to suit the circumstances and research question as well.

Analysis shown in following table reveals the flexibility of the method. Their focus, number of rounds and participants are varied from project to project

Table 1: Delphi Method Diversity

Non IS/IT Study	Delphi Focus	Rounds	No. of Experts
Gustafson, Shukla, Delbeeq & Walster	Estimate almanac events to investigate Delphi accuracy	2	4
Kuo & Yu	Identify national park selection criteria	1	28
Nambisan et al.	Develop a taxonomy of organizational	3	6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

	mechanisms		
Lam, Petri & Smit	Develop rules for a ceramic casting process	3	3
Roberson, Collins & Oreg	Examine and explain how recruitment message specificity influences job seeker attraction to organizations.	2	171
Niederman, Brancheau & Wetherbe	Survey senior IS executives to determine the most critical IS issues for the 1990s.	3	114, 126 & 104
Brancheau, Janz & Wetherbe	urvey SIM members to determine the most critical IS issues for the near future	3	78, 87 & 76
Scott	Rank technology management issues in new product development projects	3	20
Brungs & Jamieson	Identify and rank computer forensics legal issues	3	11

2.2 IMPROVED DELPHI METHOD

The classical Delphi method by four key features

1. Delphi participants anonymity - to allow participants to freely express their opinions without any pressure. Decisions are evaluated on their merit, rather than who has proposed the idea.
2. Looping process - to refine their views in light of the progress of the group's response from round to round.
3. Feedback - to clarify or change their views.
4. Statistical analysis of group response - for a quantitative analysis and interpretation of data.

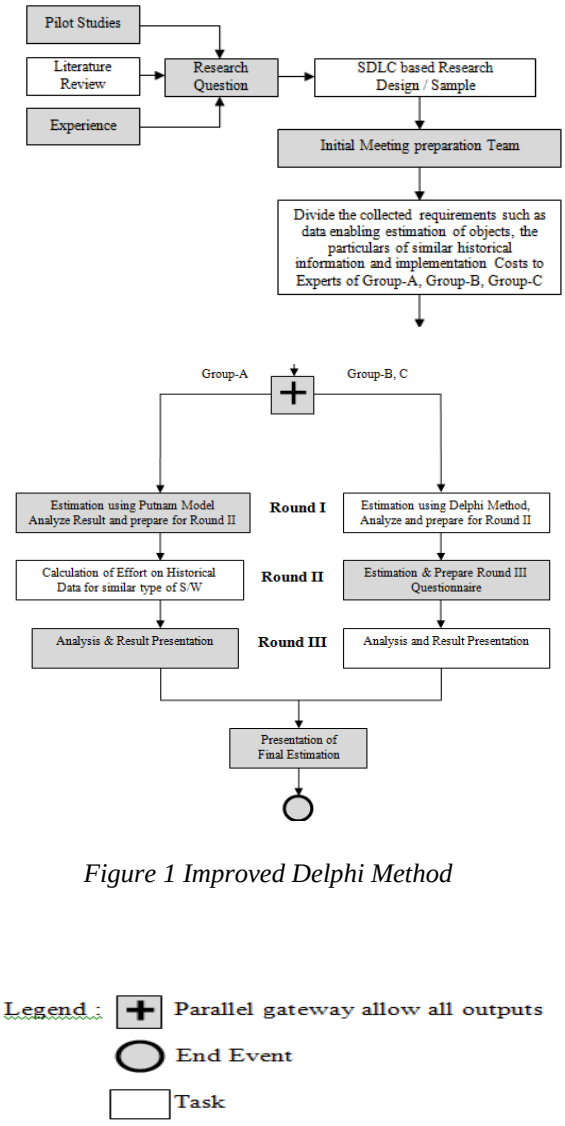

Figure 1 Improved Delphi Method

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. RELATED WORK

Estimation by expert [3][4], analogy based estimation schemes [5], algorithmic methods including empirical methods [6], rule induction methods [7], artificial neural network based approaches [8] [9] [10], Bayesian network approaches [11], decission tree based methods [12] and fuzzy logic based Estmation schemes [13] [14]. Among these diversified models, empirical estimation models are found to be possibly accurate compared to other estimation schemes and COCOMO, SLIM, SEER-SEM and FP analysis schemes are popular in practice in the empirical category [15] [16]. In case of empirical estimation models, the estimation parameters are commonly derived from empirical data that are usually collected from various sources of historical or passed projects.

Accurate effort and cost estimation of software applications continues to be a critical issue for software project managers [17]. Although expert judgment remains widely used, however, there is also increasing interest in applying statistics and machine learning techniques to predict software project effort [18] [19]. Although, neural networks have shown their strengths in solving complex problems, their limitation of being 'black boxes' has forbidden them to be accepted as a common practice for cost estimation [20].

Hardware costs, travel and training costs and effort costs are the three principal components of cost of which the effort cost is dominant [21][22]. Although many research papers appear since 1960 providing numerous models to help in computing the effort/cost for software projects, being able to provide accurate effort/cost estimation is still a challenge for many reasons. They include:

1) The uncertainty in collected measurement

2) The estimation methods used which might have many drawbacks.

3) The cost drivers to be considered along with the development environment which might not be clearly specified [23].

The most popular algorithmic estimation models include Boehm's constructive cost model (COCOMO) [24]. Thus, accurate estimation methods, for example, the FP method, have gained increasing importance [25]. The size is determined by identifying the components of the system as seen [25] by the end-user : the inputs, output, inquiries, interface [26] to other systems and logical internal files [27]. The components are classified as simple, average or complex. All these values are then scored and the total is expressed in unadjusted FPs (UFPs). Complexity factors described by 14 general systems characteristics, such as reusability [28][29], performance and complexity of processing can be used to weighed the UFP. Factors are also weighed on a scale of 0 – not present, 1 – minor influence, to 5 – strong influence [30][31]. The result of these computations is a number that correlates to system size.

Although the FP metric does not correspond to any actual physical attribute of a software system [32,33] it is useful as a relative measure for comparing projects, measuring productivity, and estimation the amount a development effort and time needed for a project [34,35]. The total number of FPs depends on the counts of distinct types of following five classes [36]. It is well documented that the software industry suffers from frequent cost overruns [37]. A contributing factor is, we believe, the imprecise estimation terminology in use. A lack of clarity and precision [38] in the use of estimation terms reduces the interpretability of estimation [39] accuracy results, makes the communication of estimates difficult and lowers the learning possibilities [40]. There are several approaches for estimating such efforts. This work proposes an improved Delphi method, using team of experts, by dividing into groups. By dividing the team of experts into groups, the developmental effort obtained is very much nearer to the planned effort and also a comparative study is done between the existing and our proposed method. The inputs are the size of the software development, a constant and a scaling factor B. The size is in units of thousands of source lines of code (KSLOC) [41]

4. DATASET DESCRIPTION

Analysis has been performed on the data set presented by Bailey and Basili [42] to develop and effort estimation model. There are three attributes in the data table, which consists of the Developed Lines of code (DLOC), the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Methodology (ME) as an element contributing to the computation of the software development effort and the measured effort. DLOC is described in Kilo Lines of Code (KLOC) and the Effort is in person-months. Attributes along with dataset is given in Table 2.

Table 2: The Dataset of NASA Software Projects

Project No.	KDLOC	ME	Actual Effort
1	90.2	30	115.8
2	46.2	20	96
3	46.5	19	79
4	54.5	20	90.8
5	31.1	35	39.6
6	67.5	29	98.4
7	12.8	26	18.9
8	10.5	34	10.3
9	21.5	31	28.5
10	3.1	26	7
11	4.2	19	9
12	7.8	31	7.3
13	2.1	28	5
14	5	29	8.4
15	78.6	35	98.7

4. EVALUATION CRITERIA

4.1 LINES OF CODE

Lines of code (LOC) also known as Source Lines of Code (SLOC) is used for measurement of the size of computer program in terms of counting number of lines in the program's source code. It is typically used to predict the amount of effort that will be required for the development of the computer software. According to Vincent Maraia [43] the SLOC values for different versions of operating systems of Windows and Linux product[44][45][46][47] are given in Table 3.

Table 3: Lines of Code for few versions of Windows and Linux Operating Systems

Year	Operating System	SLOC (Million)
1993	Windows NT 3.1	4-5
1994	Windows NT 3.5	7-8
1996	Windows NT 4.0	11-12
2000	Windows 2000	more than 29
2001	Windows XP	45
2003	Windows Server 2003	50
2009	Linux kernel 2.6.32	12.6
2010	Linux kernel 2.6.35	13.5
2012	Linux kernel 3.6	15.9
2015	Linux kernel pre 4.2	20.2

4.2 SOFTWARE EQUATION

Putnam Model describes the effort and time required to finish a software project of specified size. Managing R&D projects Putnam used his observations about productivity levels to derive the software equation:

$$\text{Effort} = \left(\frac{\text{Size}^3}{\text{Productivity} * \text{Time}^{4/3}} * B \right)$$

- where size is the software size in SLOC
- B is a scaling factor and is a function of the project size.
- Productivity is the Process Productivity, the ability of a particular software organization to product software of a given size at a particular defect rate.
- Effort is the total effort applied to the project in person years
- Time is the total schedule of the project in years.

Journal of Theoretical and Applied Information Technology
15th February 2019. Vol.97. No 3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

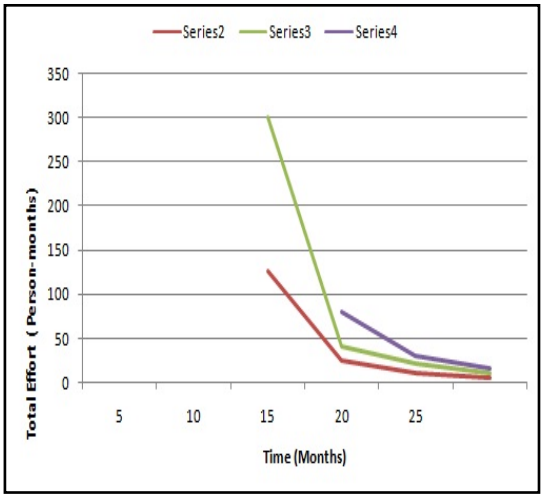

Figure 2: Plotting effort as a function of time (Time-effort Curve)

4.3 SKILL FACTOR 'B'

The special skill factor 'B' is related to the size of the product [50].

Table 4: B Value Against Size Of Software Project

B Value	Size of Software Project
0.16	5-15 K
0.18	20K
0.28	30K
0.34	40K
0.37	50K
0.39	>50K

4.4 PRODUCTIVITY 'P'

Analysis from the collected productivity data supplies initial values from variable 'P' determined by the type of software being developed. Some of the examples of various types of software's are following, but the values do not apply in all situations [44].

Table 5: P Value For Various Types Of Software's

P Value	Description
2,000	Real time embedded software
10,000	Telecommunications software
12,000	Scientific software
28,000	Business system applications

5. RESULTS AND DISCUSSION

10 projects were used from the dataset to estimate the parameters and remaining 5 projects were used for testing their performance which is shown in Table 6.

Table 6: Showing Actual Effort And Estimated Effort

Project No.	Lines of Code (KDLOC)	Actual Effort	Estimated Effort
1	90.2	115.80	117.95
2	46.2	96.00	78.56
3	46.5	79.00	73.41
4	54.5	90.80	84.24
5	31.1	39.60	43.03
6	67.5	98.40	93.47
7	12.8	18.90	22.45
8	10.5	10.30	17.19
9	21.5	28.50	35.47
10	3.1	7.00	6.93
11	4.2	9.00	11.80
12	7.8	7.30	12.58
13	2.1	5.00	3.64
14	5	8.40	9.71
15	78.6	98.70	79.84

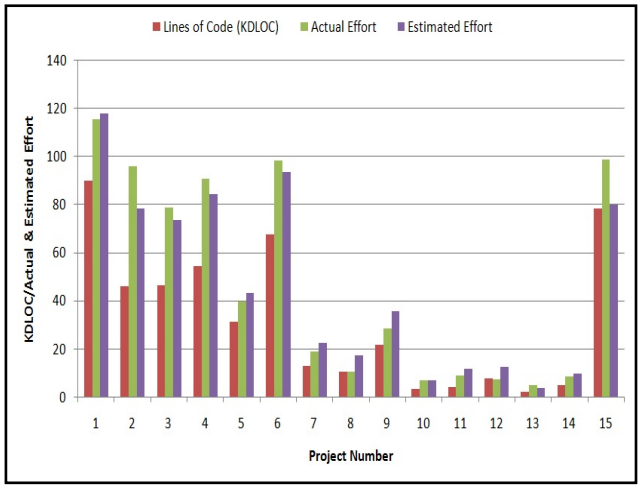

Figure 3: Plotting Actual And Estimated Effort For Various Projects

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 2 shows the graph between actual effort and estimated effort. Modified Delphi method is used to tune the parameters of software equation and the estimation capabilities are shown in Table 7.

Table 7: Models Estimation Capabilities

Model Name / Method	Model Input	Model Output	VAF
Proposed Method	size, productivity, time, b and a new biased parameter x	Effort	98.832
Model Proposed by Sheta [52]	KDLOC and ME	Effort	97.565

Table 8 shows Pred. and MMRE, therefore the proposed model has provided 39% improvement in performance and gives about 72% of projects which were predicted with a MRE less that or equals to 0.33.

Table 8: Models Estimation Capabilities

Model Name	MMRE	Pred
Proposed Model	0.2297288	72.2
Model Proposed by Sheta [52]	0.636398	38.89
% improvement	39.0	

6. CONCLUSION

The grouping of the experts into two different parts has been taken into consideration to simplify the improved Delphi process and it also reduces the probability of errors while selecting competencies during questionnaire and working on the parameters of software equation. This modified Delphi method focuses on the calculation of effort by enhancing the adjustments made to the various parameters; hence the proposed improved Delphi method ensures the quality assurance for the better effort estimation. The software size, productivity, time and scaling factors are important factors which also affects the effort and cost.

Therefore during preparation of questionnaire for various groups the parameters of software equation are taken into consideration. Because of the enhanced adjustment factor, the altered rating of the scaling factors, the effort of the software project in person month is obtained. It is found that the obtained person month is very much nearer to the planned effort and that is why this improved Delphi effort estimation technique may be recommended for the estimation of software development effort during software development process.

REFERENCES:

[1] Sheta, A. F., Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects, Journal of Computer Science 2 (2):118-123, 2006

[2] Gregory J. Skulmoski, Francis T. Hartman, Krahn J., "The Delphi Method for Graduate Research", Journal of Information Technology Education, Vol. 6, 2007, pp. 1-20.

[3] SaleemBasha, Dhavachelvan P. "Analysis of Empirical Software Effort Estimation Models" (IJCSIS) International Journal of Computer Science and Information Security, Vol. 7, No. 3, 2010.

[4] Jorgen MSjoberg D.I.K., "The Impact of Customer Expectation on Software Development Effort Estimates "International Journal of Project Management, Elsevier, pp 317-325, 2004.

[5] Chiu NH, Huang SJ, "The Adjusted Analogy-Based Software Effort Estmation Based on Similarity Distances", Journal of Systems and Software, Volume 80, Issue 4, pp 628-640, 2007.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[6] Kaczmarek J, Kucharski M, "Size and Effort Estimation for Applications Written in Java", Journal of Information and Software Technology, Volume 46, Issue 9, pp 589-600, 2004.

[7] Jefferty R, RuheM, Wieczorek I, "Using Public Domain Metrics to Estimate Software Development Effort", In Proceedings of the 7th International Symposium on Software Metrics, IEEE Computer Society, Washington DC, pp 16-27, 2001.

[8] Heiat A, "Comparison of Artificial Neural Network and Regressing Models for Estimating Software Development Effort", Journal of Information and Software Technology, Volume 44, Issue 15, pp 911-922, 2002.

[9] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort", IEEE Transactions on Software Engineering, Vol. 21, pp. 126-137, 1995.

[10] A. R. Venkatachalam, "Software Cost Estimation Using Artificial Neural Networks", Presented at 1993 International Joint Conference on Neural Networks, Nagoya, Japan, 1993.

[11] G. H. Subramaniam, P. C. Pendharkar and M. Wallace, "An Empirical Study of the Effect of Complexity, Platform and Program Type on Software Development Effort of Busness Applications, "Empirical Software Engineering, Vol. 11, pp. 541-553, 2006.

[12] R. W. Selby and A. A. Porter, "Learning from examples : generation and evaluation of decision trees for software resource analysis", IEEE Transactions on Software Engineering, Vol. 14, pp. 1743-1757, 1988.

[13] S.Kumar, B.A. Krishna and P.S. Satsangi, "Fuzzy systems and neural networks in software engineering project management", Journal of Applied Intelligence, Vol. 4, pp. 31-52, 1194.

[14] Huang SJ, Lin CY, Chiu NH, "Fuzzy Decision Tree Approach for Embedding Risk Assessment Information into Software Cost Estimation Model", Journal of Information Science and Engineering, Vol. 22, No. 2, pp.297-313, 2006.

[15] M.Van Genuchten, H.Koolen "On the use of Software Cost Models",Information & Management, Vol. 21, pp.37-44, 1991.

[16] T. K. Abdel-Hamid, "Adapting, Correcting and Perfecting Software Estimates: A maintenance metaphor", in Computer, Vol. 26, pp.20-29, 1993.

[17] K. Maxwell, L. Van Wassenhove and S. Dutta, "Performance Evaluation of General and Company Specific Models in Software

Development Effort Estimation", Management Science, Vol.45, pp. 787-803,1999.

[18] H.Azoth and R. S. D. Wahidabanu "Efficient Effort Estimation System viz. Function pointsand quality assurance coverage", IET Softw, Vol. 6, Iss. 4, pp. 335-341, 2012

[19] Deng J. D. Purvis M. K., Purvis M.A., "Software Effort Estimation: harmonizing algorithms and domain knowledge in an integrated data mining approach", Inf. Sci. Discuss. Pap. Ser., pp. 1-13, 2009 (5).

[20] Idri A., Khoshgoftaar T.M., Abran A., "Can Neural Networks be easily interpreted in software cost estimation?" 2002 World Congress on Computational Intelligence, Honolulu, Huwaii, 12-17 May 2002, pp. 1-8.

[21] Mittal H., Bhatia P., "A comparative study of conventional effort estimation and fuzzy effort estimation based on triangular fuzzy number", Int. J. Comput. Sci. Secur., 2002, 1, (4), pp. 36-47.

[22] Benton A., Bradly M. "The International Function Point User Group (IFPUG), in Funtion point counting practices manual – release 4.1 (SA. 1999).

[23] Aljahadi S., Sheta A. F., "Software Effort Estimation by tuning COCOMO model parameters using differential evolution", Int. Conf. on Computer Systems and Applications (AICCSA), 16-19 May 2010, pp. 1-6.

[24] Boehm B. W., "Software Engineering Economics", Prentice Hall, Englewood Cliffs, NJ, 1981.

[25] Peischl B., Nica M., Zanker M., Schmid W., "Recommending Effort Estimation Methods for Software Project Management", Proc. IEEE/WIC/ACM Int. Conf. on Web Intelligence and Intelligent Agent Technology, Milano, Italy, 2009, Vol. 3, pp. 77-80.

[26] B. W. Boehm, "Software Engineering Economics", Prentice Hall, 1981.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[27] B. W. Boehm, E. Horowitz, R. Madachy, D. Reifer, B. K. Clark, B. Steece, A. W. Brown, S. Chulani and C. Abts, "Software Cost Estimation with COCOMO II", Prentice Hall, 2000.

[28] F. J. Heemstra, "Software Cost Estimation", Information and Software Technology, Vol. 34, pp. 627-639, 1992.

[29] N. Fenton, "Software Measurement: A necessary Scientific Basis", IEEE Transactions on Software Engineering, Vol. 20, pp. 199-206, 1994.

[30] BarryBoehm, Chris Abtsa and Sunita Chulani, "Software Development Cost Estimation Approaches ~ A Survey", Annals of Software Engineering, pp. 177-205, 2000.

[31] James Nelson H., Monarchi D. E., "Eknsuring the quality of conceptual representations", Softw. Qual. J., 1997, 15, (2), pp. 213-233.

[32] Khoshgoftaar T. M., Allen E. B., Naik A., Jones W. D., Hudepohl J. P., "Using Classification trees for software quality models : Lesssons learned", Int. J. Softw. Eng.Knowl. Eng., 1999, 9, (2), pp. 217-231.

[33] Kitchenham B. A., "Cross versus within – company cost estimation studies: a Systematic review" IEEE Trans. Software Eng., 2007, 33, (5), pp. 316-329

[34] Hannay J. E., Sjoberg D. I. K., Dyba T., "A systematic review of theory use in Software Engineering Experiments", Softw. – Pract. Exper., 2007, 33, (2), pp. 87-107.

[35] Jack E. Matson, Bruce E. Barrett and Joseph M. Mellichamp, "Software Development Cost Estimation Using Function Points", IEEE Transactions on Software Engineering, Vol. 20, No. 4, April 1994.

[36] Chris F. Kemerer "An Empirical Validation of Software Cost Estimation Models".

[37] Putnam L. H., "General empirical solution to the macro software sizing estimating problem", IEEE Trans. Softw. Eng. SE 4, 4 (July 1978), pp. 345-361.

[38] Putnam L. and Fitzsimmons A., "Estimating Software Costs", Datamation 25, lo-12 (Sept. - Nov. 1979)

[39] B. Boehm, C. Abts and S. Chulani, "Software Development Cost Estimation Approaches – A Survey", Technical Report USC-CSE-2000-505, "University of Souther California – Center for Software Engineering, USA (2000).

[40] Chulani S., Boehm B. and Steece B., "Bayesian Analysis Emperical Software Engineering Cost Models", IEEE Trans. Software Eng., vol. 25, no. 4, pp. 573-583, 1999.

[41] Barry Boehm, "COCOMO II Model Definition Manual", Version 1.4 – Copyright University of Southern California.

[42] Bailey, J. W. and V. R. Basili, 1981. A meta model for software development resource expenditure. Proc. Intl. Conf. Software Engineering, pp: 107-115.

[43] "How Many Lines of Code in Windows?". Knowing.NET. December 6, 2005. Retrieved 2010-08-30

[44] "What's new in Linux 2.6.32". Archived from the original on 2013-12-19 Retrieved 2009-12-24

[45] Greg Kroah-Hartman; Jonathan Corbet; Amanda McPherson (April 2012). "Linux Kernel Development: How Fast it is Going, Who is Doing It, What They are Doing, and Who is Sponsoring It" The Linux Foundation. Retrieved 2012-04-10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

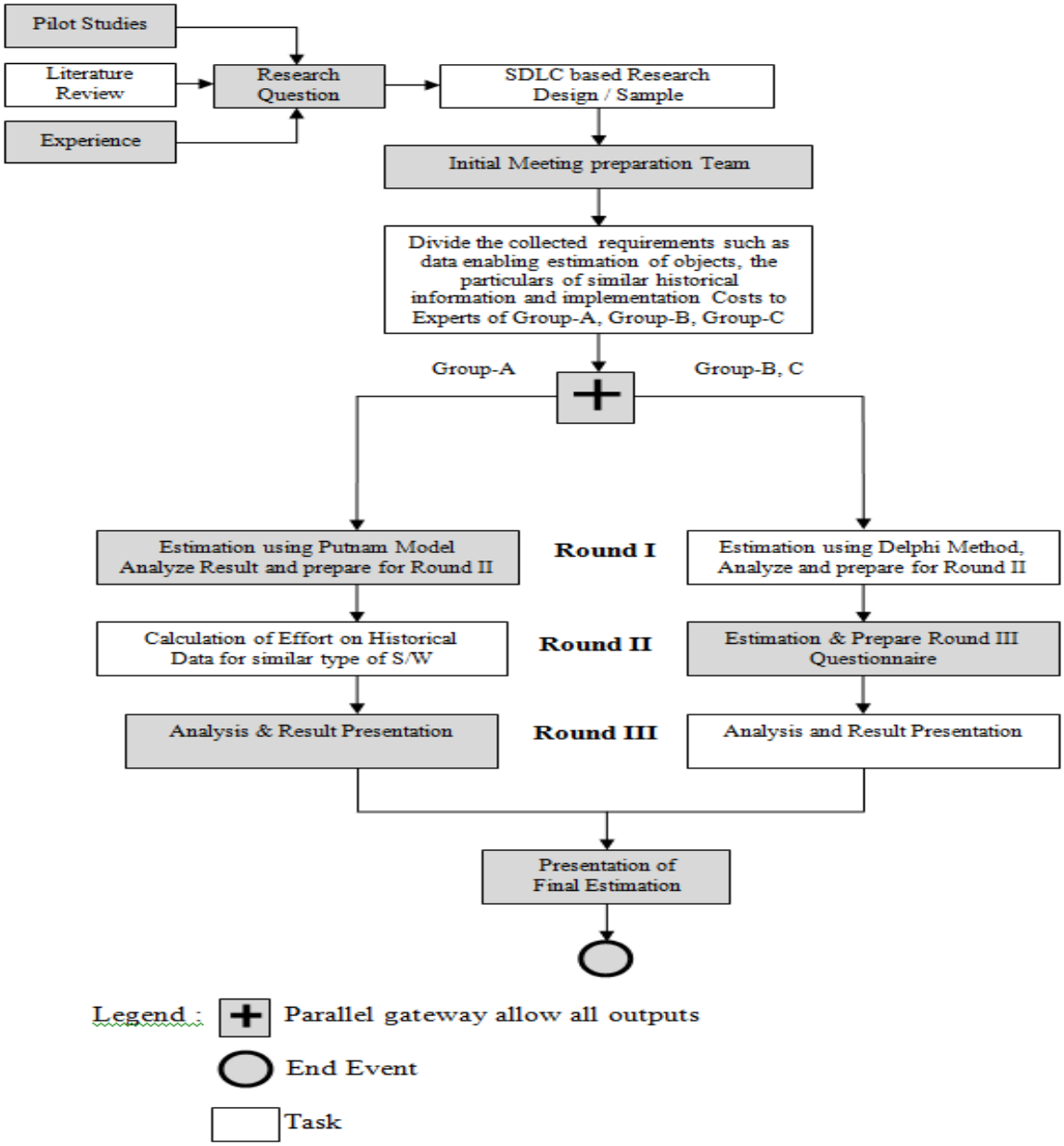

Figure 1 Improved Delphi Method