A Double-Weighted Parametric Model for Academic Software Project Effort Estimation

Jatinderkumar R. Saini and Vikas S. Chomal

Abstract The effort, cost, and time play a vital role in the success or failure of the software. The ratio of software project failure nowadays is growing like a storm in the world. One of the reasons behind this failure proportion is an imprecise and inappropriate estimate of required effort, cost, and budget for particular software project development. The motivation behind our research is to estimate the effort for software development accurately. Accurately estimate effort for software development is one of the most challenging tasks because from a very early stage it requires in-depth study as well as detailed statistics. Participation of members of the team is important during the development of the project along with various activities like domain area, sector, nature, software process methodology, and other resources. All these proofs do not appear in the initial phase of development, but on the other side, they discovered as software growth progresses. Many models proposed for estimating effort related to software development, but only a few are adaptable for effort estimation task in the academic software project. From the study of nearly 600 academic software projects and inputs from more than 30 Software Engineering experts, this research paper proposes an innovative model based on 8 parameters and 24 sub-parameters. Each of the sub-parameters is weighted twice, and the final effort estimation obtained by summation of the individual product of the weight of the parameter and weight of sub-parameter. We have coined the term "ASPEE units" (Academic Software Project Effort Estimation) for the estimated effort.

Keywords Educational perspective · Software engineering · Software estimation models · Estimate effort · Software project

J. R. Saini
Symbiosis Institute of Computer Studies and Research, Pune, Maharashtra, India
e-mail: saini.expert@gmail.com

V. S. Chomal (✉)
TMES Institute of Computer Studies, Mandvi, Surat, Gujarat, India
e-mail: vikschomal80@gmail.com

S. Fong et al. (eds.), *ICT Analysis and Applications*, Lecture Notes in Networks and Systems 93, https://doi.org/10.1007/978-981-15-0630-7_4

1 Introduction

One of the oldest definition of Software Engineering (SE) states that "it is a creation and utilization of comprehensive engineering principles to acquire software which is consistent, cost-effective as well as function competently on machines." Boehm states that "software engineering makes use of science and mathematics through which the competencies of computer equipment are provided to the user using computer programs, procedures and associated documentation" [1]. Haapio defines software engineering engagement of real people in real environments [2]. The software is shaped, conserved, and advanced by people. Apart from technical activities, SE also incorporates managerial activities such as defining roles and responsibilities, organizing the workflow, scheduling and tracking the progress of work down as well as synchronization of individuals working to fulfill a common objective [3]. The software managerial aspects focus on planning and estimation of software development time, resources required as well as cost and effort required. Software development transits through different phases as mentioned in the Software Development Life Cycle (SDLC) such exploration of requirements, design, code, test, documentation, and maintenance rigidly followed. Apart from these conventional stages, one of the most important as well as challenging tasks is to estimate the effort required in these various phases of SDLC accurately. If effort in software development is not measured correctly, calculated and followed, then it may result in quality failure, or even it may result in a complete failure of the software.

According to Haapio [2], the term effort in software development can define as the collective amount of time that is consumed by the project, and each project required a specific number of persons per hours, days, months, or years which is directly associated with the size of the project. Hence, the effort is the product of people and time which can be expressed as effort = people * time, whereas, estimation can be defined as a probabilistic valuation with a rational and precise value of the center of a range. Estimation can also be considered as a prediction. Therefore, all existing estimation model can be viewed as a prediction model also. For project scheduling, tracking, and controlling appropriate estimation is to be done and to make a real and effective estimation for any project is a challenging task. Estimation with software project development includes—(a) Estimation of size, (b) Estimation of Effort, (c) Estimation of Cost, and (d) Estimation of Schedule.

In effort estimation many existing models and techniques are available. Some of them are—(a) Empirical Parametric Estimation Model such as COCOMO, Putnam's Software Life Cycle Model (SLIM), (b) Empiric Non-parametric Estimation Model such as Optimized Set Reduction Technique (OSR), Decision-Making Trunk, (c) Expert Estimates such as Delphi technique, (d) Analogue Estimation Models such as ESTOR and ANGEL, (e) Downward Estimates, and (f) Upward Estimates. Software project development is a compulsory requisite for obtaining a master's degree in each computer science stream. The ultimate standard behind this is to make the students identify various factors that have a consequence on software development. During

their software project development, students are required to focus on the following aspects of the software:

1. Requirement Analysis
2. System Design
3. Database Design
4. Structural and Object-Oriented Modeling Techniques
5. Coding
6. Testing

These six phases are extremely imperative as well as should be firmly and appropriately followed by students while developing their software within a stipulated time-bound. One of the collective opinions of academicians is that apart from regular monitoring and guidelines provided during the project work, the majority of students' projects remain incomplete. There may be many reasons behind this cause such as students fail to identify the nature and scope of software to be developed, requirements are not properly gathered, prioritized, and understood, having ambiguity in the functionality of the software and so on. It also perceived that students do not give crucial significance to recognize the effort required in every phase of software development. Looking at all these difficulties, an effort estimation model proposed which can utilize for estimating the effort in student's software project in the computer science curriculum. The rest of the paper is organized as follows. In Sect. 2, the related literature review presented, supported by systematic research methodology conducted during our study in Sect. 3. Section 4 represents experiments and results followed by the conclusion in Sect. 5.

2 Literature Review

Hathaichanok and Nakornthip proposed a framework which was used to estimate software cost and effort [1]. The proposed model based on the relational matrix utilizing the principle of multiple regression analysis and analogy method. J. Hill et al. studied the accuracy and effectiveness of experts' subjective estimates for the tasks that are carried out in software project development [4]. In their experiment, they concluded that the majority of functions were found to be overestimated. Further, Jovan et al. gave detail clarification as well classification of various effort estimation models and techniques used [5]. In their report, they accomplished that even though many prevailing models and methods then also effort estimation remains unpredictable. Jyoti and Vikas systematically represent a brief analysis of software effort and cost estimation [6]. They also conducted an experiment showing the performance of various estimation models and techniques. Kassem provided with multiple guidelines and checkpoints considered for effort and cost allocation and estimation for particular software development activities [7]. The existing requirements based estimation techniques used by the authors for conducting this study. Kjetil and Magne

have proved that most of the software projects face either effort or schedule overruns [8]. During the study, they summarize that there is inadequate evidence of why this effort and schedule overruns occur. Lava in their research study chiefly considered effort estimation needed in testing the software been developed [3]. A survey and interview procedure was conducted, and the authors reveal that the majority of companies prepare separate estimation for testing phase and this estimation strictly associated with overall development effort. Jorgensen worked out with expert estimation technique and provided various guidelines based on expert estimation [9]. Mohammed and Rizwan presented an in-depth comparative study on software effort estimation models [10]. They also proposed a model that can be useful for information technology organizations for estimating efforts, budget, and in the decision-making process. Mudasir presented a review of various effort estimation and also suggested various advantages of models [11]. Muhammad et al. developed and provided with multiple checklists considering agile methodology and suggested various improvements made in effort estimation for agile methodology [12]. Paweł et al. proposed a course that was considered for understanding effort estimation in student's software project development [13]. Suri and Pallavi encapsulate various cost and effort estimation models and techniques [14]. They also implement simulators in their study and observed that no single technique best suited to given situations.

Poonam et al. focus on how various parametric and non-parametric estimation models used and what are their effects [15]. Putu and Sholiq considered Use Case Points popularly known as UCP in their exploration [16]. They made UCP as a base and premeditated amount of effort estimation required in a software development project. Reetuparna and Taniya described the various phase effort distribution patterns and causes of their variation. In their experiment, they found that the pattern shows consistency for phases like testing, as well as consistency in the effects of software size and team [17]. But a large amount of variation was observed in phases like design and requirement gathering. Basha and Dhavachelvan presented a detailed study of various software effort estimations [18]. Sangeetha and Pankaj investigated and presented a comparative study about different estimation techniques and also a detailed illustration of the model which makes use of Line of Code (LOC) and Function Point (FP) was presented [19]. Sarah and Maqbool proposed a model that can utilize to estimate effort for every task of SDLC [20]. Instead of relying on cost and line of code, they considered individual task as the primary factor while implementing their model. Thomas gave a domain-based effort distribution model that can help in estimating resource allocation for a particular domain [21]. Ursula and Shepperd conducted an experiment considering students' project with various checklists and group discussion [22]. The purpose was to improve the estimation. The result of the analysis shows that there was a significant improvement in estimation while applying these checklists and guidelines. Vishakha and Gaurav proposed a model for commercial software organizations [23]. Their model used to estimate and analyze various attributes of a software project like complexity, size, effort, development team as well as budget. This model can be used during the initial phase of software development to estimate these factors. Werner and Michel defined that effort is a

term that consumed during software development considering the time duration of each phase of SDLC [24].

3 Methodology

The research initially commenced with an exploration of the literature review related to software effort estimation. The next phase includes exploration and investigation of various stages followed by students during software project development. For this investigation, we considered 589 software project documentations of final year students of the master's degree course. The duration time of these software project developments was six months. At the same time, all these projects were considered as full-time projects for obtaining a master's degree under computer science. The documentation was collected from the college library. After scrutinizing these massive software project documentations, we disclosed the following phases followed during their software project development. They are (a) Learning and exploring the technology which students utilized for software development, (b) Gathering and Analysis of software requirements, (c) Designing and Implementation of software, and (d) Testing. The various phases of the research task carried by us is presented in Fig. 1.

Further, the subsequent stage followed was listing and identification of parameters that are proportionally significant for estimating effort for software project development and explained in Table 1.

Further, these parameters are the deliberate perspective of estimating effort for software project development and not in the context of projects. After listing and clarification of these parameters, the next stage is to score these parameters. Scoring to these parameters was done based on the importance of each parameter in effort estimation for student software project development. It also followed that the total of these parameters should sum up to 100. For scoring these parameters, a survey was conducted with 31 academicians teaching in graduate and undergraduate colleges of the computer stream. All academicians fulfilled the criteria of holding at least the master's degree with first class and an experience of at least 10 years in the academic domain. Experiment and observations are presented in the next section.

4 Experiments and Results

The result of the survey is presented in Table 2 as well as graphically in Fig. 2a, b. The Column "CSA" in Table 2 specifies for the score provided by nth Computer Science Academicians with ranging from 1 to 31.

Based on the sum of parameters of the proposed effort estimation model values presented in Fig. 2a, it has been found that "Technology" scored the maximum sum (530) whereas "Experience" achieved the second highest score (475). The "Domain" was found to achieve the third highest score (450). Similarly, the minimum sum was

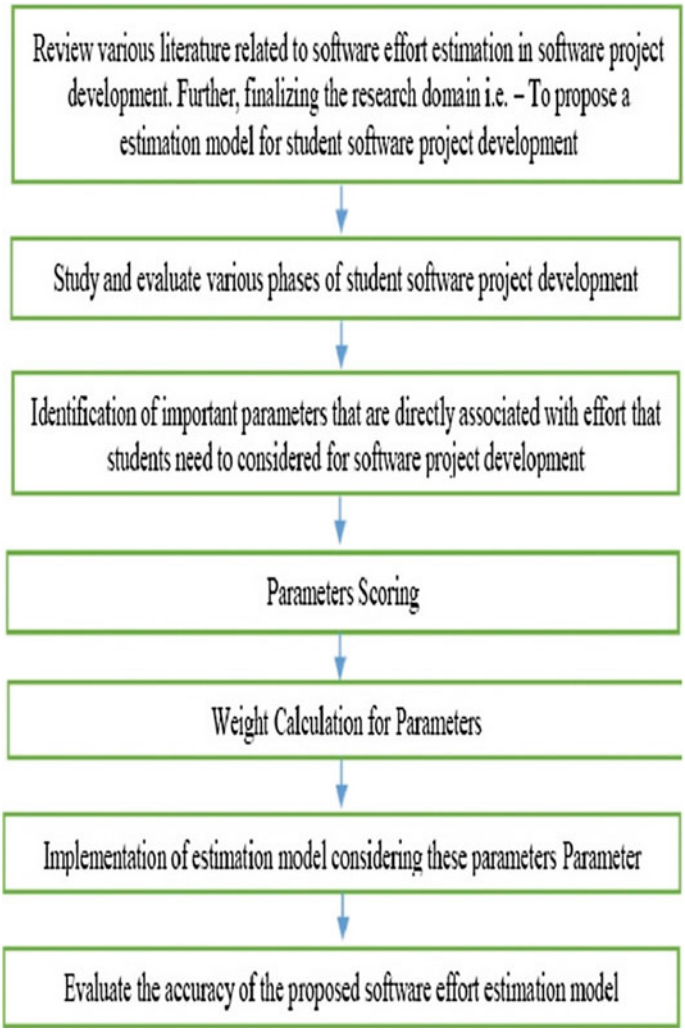

Fig. 1 Diagrammatic representation of research methodology

found to be assigned to "Team Size" (190) while the second lowest sum was found to be achieved by "Parallel Subject" (215).

Column Chart as depicted in Fig. 2b is used to represent clarification of the average values presented in Table 2, where on X-axis parameters of the proposed model are highlighted and on Y-axis average assigned are shown. From Fig. 2b it has been observed that maximum average was assigned to Technology (17.10%), while Experience with (15.32%) was found to have achieved the second highest whereas Domain was third highest with (14.52%). Similarly, the minimum average was found to be assigned to Team Size (6.29%) while the second lowest was found

Table 1 The parameters of the proposed model

Sr. no.	Parameters	Description
1	Project type	The type of project to be developed. For example, Desktop-based, Web-based portal
2	Technology	The technology used for developing software
3	Team size	The number of members involved in developing software
4	Domain	Application area for which software is developed. For example, Principle-based, Corporate- and Commercial-based, Decision-based, Network- or Security-based, Scientific-based, Social Networking-based
5	Project Duration	Time duration of software project development
6	Experience	The expertise of team members involved in software development. Having any prior knowledge of developing software
7	Functionality-based project complexity	The complexity of software to be developed
8	Parallel subject(s)	Any parallel subject to be learnt by the student along with software project development

to be achieved by Parallel Subject (6.93%). The next step is the calculation of weight for parameters identified and listed. The formula used for calculation of weight is:

$$\text{Weight} = (\text{Average}\ (\%)/100) \tag{1}$$

The value of variable Average (%) in formula (1) derived from Table 2. Further, the weight calculation for each parameter presented in tabular format in Table 3.

For better clarification regarding weight assignment to each parameter of the proposed model represented pictorial using a column chart in Fig. 3 where parameters plotted on the X-axis, and Weight was shown on Y-axis. From the Fig. 3, it is found that computer science academicians gave maximum weight to "Technology" (0.17), while "Experience" (0.15) was found to have achieved the second highest weight. Similarly, the minimum weight was found to be assigned to "Team Size" (0.062) while the second lowest weight was found to be achieved by "Parallel Subject" (0.069). Further, the next step was to provide weight to specific attributes of the main eight parameters listed in Table 1. The corresponding weightage presented in Table 4. At this stage, we are assigning weights to sub-parameters of parameters which have already assigned weights. Hence, we call our model a double-weighted

Table 2 Survey result

Sr. no.	Parameters	Respondent's response						Sum	Average (%)
		CSA1	CSA2	CSA3	CSA4	CSA5	…CSAn		
1	Project type	10	10	15	10	15	…..	415	13.39
2	Technology	20	20	15	10	5	…..	530	17.1
3	Team size	5	5	5	5	10	…..	195	6.29
4	Domain	20	20	10	15	10	…..	450	14.52
5	Project duration	5	5	10	15	20	…..	410	13.23
6	Experience	20	20	20	10	15	…..	475	15.32
7	Functionality-based project complexity	15	15	15	15	10	…..	410	13.23
8	Parallel subject(s)	5	5	10	20	15	…..	215	6.935
Total		100	100	100	100	100	…..	3100	100

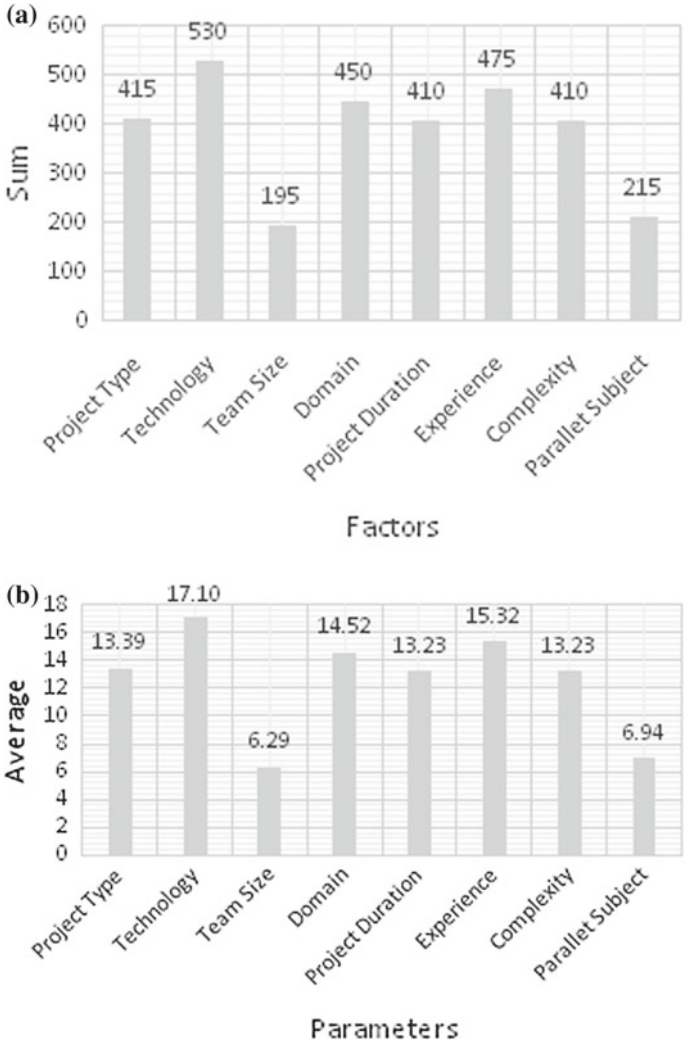

Fig. 2 **a** Sum of parameters of proposed effort estimation model. **b** Average of sum of parameters for proposed effort estimation model

parametric model. If the weight of a parameter is "n", then the product of the weight of its sub-parameter with "n" increases the weightage n-times.

In Table 4, the eight parameter(s) further characterized to sub-parameter (attribute/characteristics). For example, desktop, web, and portal are sub-parameter of "Project" type parameter. Our next step is to determine the impact of all these sub-parameters in estimating effort. In software development, the "attribute impact" consists of three group values with weight based on effect and complexity, e.g., (a). Low, Medium, and High with (1, 2, 3). (b). Yes, and No with (1, 2). (c). Lowest,

Table 3 Weight calculation

Sr. No.	Parameters	Weight
1	Project type	0.134
2	Technology	0.171
3	Team size	0.063
4	Domain	0.145
5	Project duration	0.132
6	Experience	0.153
7	Functionality-based project complexity	0.132
8	Parallel subject(s)	0.069

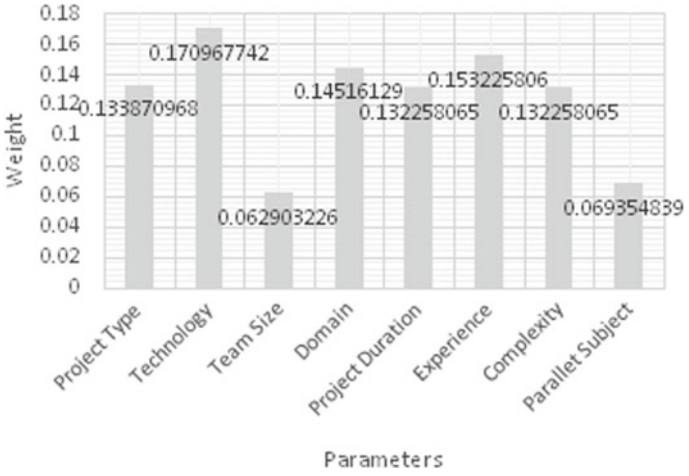

Fig. 3 Weight to factors of proposed effort estimation model

Lower, Low, High, Higher, and Highest with (1, 2, 3, 4, 5, 6). As each parameter and corresponding sub-parameters are unique and different from each other, it is noteworthy that this is repeated for each of the eight parameters under study. Now, the next step is to calculate the score for each parameter for estimating effort estimation. The formula mentioned below, and a unit considered in the formula termed as Academic Software Project Effort Estimation abbreviated as ASPEE (units), which is our coined term. For calculation of ASPPE units, the weight assigned to each parameter and attributes of parameters which represented in Tables 3 and 4, respectively are considered and mentioned below.

$$\text{Parameter(s)\_Weight} * \text{Attribute(s)\_Weight} \tag{2}$$

For example, let us consider project type as a parameter having weightage value (0.13) and desktop as parameter attributes which is having weight (1) since its effort

Table 4 Weight calculation for attributes of parameter(s)

Sr. no.	Parameter(s)	Attributes/Characteristics	Attribute Impact	Weight (Based on effort and complexity)
1	Project type	Desktop	Low	1
		Web	Medium	2
		Portal	High	3
2	Technology	Aware	Yes	1
		Not aware	No	2
3	Team size	1 Member	High	2
		2 Members	Low	1
		More than 2 Members	High	2
4	Domain	Scientific-based	Highest	6
		Principle-based	Higher	5
		Network or Security based	High	4
		Decision-based	Low	3
		Corporate and Commercial	Lower	2
		Social networking	Lowest	1
5	Project duration	6 Months	Low	1
		4 Months	Medium	2
		2 Months	High	3
6	Experience	Prior development experience	Yes	1
		No experience	No	2
7	Functionality-based project complexity	Less functionality required	Low	1
		Moderate functionality required	Medium	2
		Too much functionality required	High	3
8	Parallel subject(s)	No	Low	1
		Yes	High	2

and complexity is considered as low, hence the calculation of effort for this parameter will be:

$$0.13(\text{Parameter\_Weight}) * 1(\text{Attribute\_Weight}) = 0.13 \text{ ASPEE units} \quad (3)$$

Similarly, the calculation of effort for all 8 major parameters namely Project Type, Technology, Team Size, Domain, Project Duration, Experience, Functionality-based

Project Complexity, and Parallel Subject(s) as stated in Table 4 needs to be calculated. Finally, a summation of effort estimation for all parameters is done to yield the final effort estimate for the entire project. For experimentation of our proposed model, software project documentations of 10 groups were considered for calculating effort estimation. The experiment and relevant results presented in Table 5.

In Table 5, a total of ASPEE units are presented for each project. Further, if there is a change in the weights of the specified parameters such as technology, team size, and functionality-based project complexity or any other parameters, then there will be some deviations in total ASPEE units. Hence these parameters can also be termed as quantifiable parameters and comparison can be defined as a quantifiable comparison. The primary goal of formulating ASPEE units is to calculate and present the estimated effort for student's software project development.

5 Conclusion and Future Work

Today's digital world experiences swift variation in technologies and due to this software development in this period is at a challenging stage, and estimation of effort to be dedicated in the implementation of software remains complex as well as a challenging task. Further, the processes executed during software development and effort estimation are strongly cohesive. From the review, literature examined and observed that effort estimation in software development always remains a promising domain among the research community. During a period of the span, many diverse models and effort estimation methods were introduced and implemented. All these undoubtedly designate the awareness among the scholars of the need to improve effort estimation in software engineering. Practically, all IT companies globally consider effort estimation while developing a software project. But it is commonly not considered or unnoticed in developing academic software projects by students of computer science courses. Academic software projects form a mandated part of the curriculum for completion of a course. Often, as the project development progresses, various obstacles such as lack of technology-knowledge, the complexity of functionalities, pressure due to parallel subjects, etc. are experienced by the developer students. Considering the same, we believe that academic domain dealing with software project development oriented courses should focus, consider, and provide guidelines as well as approaches, models regarding effort estimation in software project development.

Our research also focuses on effort estimation with the significant objective of proposing a conceptual model for estimating the effort required for software project development in academic dominion. There are three primary stakeholders involved in the academic software project: Students, College/University Supervisor, and Company/Organization Supervisor. Given many projects and in lack of any quantified estimated value for expected effort, it often becomes difficult for either supervisor to assign a group of students to a specific project. This results, at times, in a difficult project assigned to a weak group of students. In any case, the project assignment is often done randomly and just based on intuitive guts rather than a quantified estimate

Table 5 Result of experimentation

Sr. no.	Project title	Parameters								Total (ASPEE Units)
		1	2	3	4	5	6	7	8	
		Estimate values								
1	Inventory management system	0.134	0.171	0.063	0.435	0.264	0.153	0.264	0.069	1.553
2	Online grocery store	0.268	0.171	0.126	0.29	0.264	0.153	0.396	0.069	1.737
3	Milk distribution	0.402	0.171	0.126	0.435	0.264	0.153	0.396	0.069	2.016
4	Lakshya blood bank	0.268	0.171	0.063	0.29	0.264	0.153	0.132	0.069	1.41
5	Network management system	0.268	0.171	0.126	0.58	0.264	0.153	0.396	0.069	2.027
6	Material management system	0.402	0.342	0.063	0.435	0.264	0.153	0.396	0.069	2.124
7	Accounting tracking system	0.268	0.171	0.126	0.725	0.264	0.153	0.396	0.069	2.172
8	Online food ordering system	0.268	0.171	0.063	0.29	0.264	0.153	0.132	0.069	1.41
9	A to Z GIS Map App	0.268	0.342	0.126	0.58	0.264	0.153	0.396	0.069	2.198
10	Online shoe store	0.268	0.171	0.126	0.29	0.264	0.306	0.132	0.069	1.626

for the effort. This research work has presented an effort estimation model for such academic software projects and claims that there will be a high success rate of project completions and completions on time and with fewer errors. In our research work, we identified 8 parameters and 24 sub-parameters. These individual sub-parameters are weighted twice, and the final effort estimation obtained by summation of the individual product of the weight of parameter and weight of sub-parameter. Further, we coined the term "ASPEE units" (Academic Software Project Effort Estimation) for the estimated effort. The fundamental goal of assigning double weightage to these parameters and to compute ASPEE units is to implement a model in the future that will be used to calculate effort estimation for academic software projects.

References

1. Suwanjang H, Prompoon N (2012) Framework for developing a software cost estimation model for software modification based on a relational matrix of project profile and software cost using an analogy estimation method. Int J Comput Commun Eng 1(2)
2. Haapio T (2011) Improving effort management in software development projects. Publications of the University of Eastern Finland Dissertations in Forestry and Natural Sciences
3. Kafle LP (2014) An empirical study on software test effort estimation. Int J Soft Comput Artif Intel 2(1). ISSN: 2321-404X
4. Hill J, Thomas LC, Allen DE (2000) Experts' estimates of task durations in software development projects. Int J Project Manag 18. www.elsevier.com/locate/ijproman
5. Zivadinovic J, Medic Z, Maksimovic D, Damnjanovic A, Vujcic S (2011) Methods of effort estimation in software engineering. In: International symposium engineering management and competitiveness 2011 (EMC2011) June 24–25, 2011, Zrenjanin, Serbia
6. Borade JG, Khalkar VR (2013) Software project effort and cost estimation techniques. Int J Adv Res Comput Sci Softw Eng 3(8). ISSN: 2277 128X
7. Saleh K (2011) Effort and cost allocation in medium to large software development projects. Int J Comput 5(1)
8. Molokken K, Jorgensen M (2003) A review of surveys on software effort estimation. In: ISESE'03 proceedings of the 2003 international symposium on empirical software engineering
9. Jorgensen M (2004) A review of studies on expert estimation of software development effort. J Syst Softw 70:37–60. www.elsevier.com/locate/jss
10. Aljohani M, Qureshi R (2017) Comparative study of software estimation techniques. Int J Softw Eng Appl (IJSEA) 8(6)
11. Kirmani MM (2017) Software effort estimation in early stages of software development: a review. Int J Adv Res Comput Sci 8(5). ISSN No. 0976-5697
12. Usman M, Petersen K, Börstler J, Neto PS (2018) Developing and using checklists to improve software effort estimation: a multi-case study. J Syst Softw. ISSN 0164-1212, E-ISSN 1873-1228
13. Skruch P, Długosz M, Mitkowski W (2017) Improving the success rate of student software projects through developing effort estimation practices. In: Springer International Publishing AG 2017, Trends in advanced intelligent control, optimization and automation, advances in intelligent systems and computing, p 577. https://doi.org/10.1007/978-3-319-60699-6_83
14. Suri PK, Ranjan P (2012) Comparative analysis of software effort estimation techniques. Int J Comput Appl 48(21) (0975–8887)
15. Rijwani P, Jain S, Santani D (2014) Software effort estimation: a comparison based perspective. Int J Appl Innov Eng Manag 3(12)

16. Primandari PL, Sholiq (2015) Effort distribution to estimate cost in small to medium software development project with use case points. In: The third information systems international conference, procedia computer science, vol 72, pp 78–85. www.sciencedirect.com
17. Mukherjee R, Gupta T, Thirugnanam M (2016) Review of effort distribution in IT companies. Int J Eng Techn 2(6)
18. Basha S, Ponnurangam D (2010) Analysis of empirical software effort estimation models. Int J Comput Sci Inf Secur 7(3)
19. Sangeetha K, Dalal P (2015) A review paper on software effort estimation methods. Int J Sci Eng Appl Sci (IJSEAS) 1(3). ISSN: 2395-3470
20. Safavi SA, Shaikh MU (2011) Effort estimation model for each phase of software development life cycle. www.igi-global.com/chapter/effort-estimation-model-each-phase/46269, https://doi.org/10.4018/978-1-61520-789-3
21. Tan T (2012) Domain-based effort distribution model for software cost estimation. A Dissertation Presented to the Faculty Of The Usc Graduate School University Of Southern California In Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy (Computer Science)
22. Passing U, Shepperd M (2003) An experiment on software project size and effort estimation. In: IEEE Xplore conference: empirical software engineering. https://doi.org/10.1109/isese.2003.1237971
23. Sharma V, Garg G (2017) Usage and analysis of estimation techniques in software project management. Int J Innov Res Comput Commun Eng 5(5). ISSN (Online): 2320-9801 ISSN (Print): 2320-9798
24. Heijstek W, Chaudron MRV (2007) Effort distribution in model-based development. In: Proceedings of the 2nd workshop on model size metrics (Co-located with MODELS 2007) Nashville, TN, USA