Advances in Engineering Software 175 (2023) 103329

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Research paper

Prediction based cost estimation technique in agile development

Shariq Aziz Butt[a,*], Tuncay Ercan[b], Muhammad Binsawad[c], Paola-Patricia Ariza-Colpas[d], Jorge Diaz-Martinez[d], Gabriel Piñeres-Espitia[d], Emiro De-La-Hoz-Franco[d], Marlon Alberto Pineres Melo[d,e], Roberto Morales Ortega[d], Juan-David De-La-Hoz-Hernández[e]

[a] *Department of Computer Science, University of South Asia, Lahore, Pakistan*
[b] *Department of Management Information Systems, Yasar University, Turkey*
[c] *Department of Computer Information Systems, King Abdulaziz University, Jeddah, Saudi Arabia*
[d] *Department of Computer Science and Electronics, Faculty of Engineering, Universidad de la Costa, Barranquilla, 080002, Colombia*
[e] *Department of Systems Engineering, Universidad del Norte, Barranquilla, 081001, Colombia*

ARTICLE INFO

Keywords:
Agile limitations
Agile project cost and time estimation
Estimation technique

ABSTRACT

Agile has been invented to improve and overcome the deficiencies of efficient software development. At present, the agile model is used in software development vastly due to its support to both developers and clients resourcefully. Agile methodology increases the interaction between the developer and client to make the software product defect-free. The agile model is getting to be a well-known life cycle model because of its particular features and most owing is to allow changes at any level of the project from the product owner. However, on other hand, this novel feature is a disadvantage of the agile model due to frequent change requests from the client has increased the cost and time. To overcome cost and time estimation issues different cost estimation techniques are being used in agile development but no one is pertinent for accurate estimation. Therefore, this study has proposed a cost estimation technique. The proposed estimation technique is predictions-based and has different categorizations of projects based on user stories complexities and the developer's expertise. We have applied the suggested technique to ongoing projects to find the results and effectiveness. We have used two projects with different sizes and user stories. Both projects have different modules and developers with different expertise. We have used the proposed estimation technique on projects and done a survey session with the teams. This survey session's main objective is to reveal the statistical findings of the proposed solution. We have designed the 12 hypotheses for statistical analysis.

1. Introduction

The agile software development model is new and essentially being used in software development. It enables clients to make change requests at any time during the project [1, 2]. As a result, module coordination is prevalent within software companies. The agile paradigm, in addition to having a great number of benefits and being widely used, also has a few drawbacks. One of these drawbacks is the most visible factor which constantly modifies the client's requirement, causing the task's completion time and cost to increase. The expense and time increase likewise impacts the industry image on the client. The company can lose its client and the client resembles a resource for the organization. There are cost and time estimation systems accessible to COCOMO I yet this procedure isn't valuable to understanding the issue from agile development improvement. In this way, there is a need for a progressively indigenous and precise estimation strategy or model [3–5]. The author targets only the industries that are using the agile model for software development. With this line of research, we have done the research. This paper is an extended version of our previously published paper in a journal. In our previous publication, we had done a survey based on hypotheses, interview sessions, and meetings with development teams. The team includes developers, project managers, and software architects. In the last paper, we had done a comparison with the existing techniques and in a survey, we got responses from the participants regarding the estimation technique features, reliability, and accuracy to estimate efforts. However, in the extended version, we have visited software industries to apply the cost estimation technique to projects [6–8]. The main difference between the previous version and

* Corresponding author.
E-mail address: shariq2315@gmail.com (S.A. Butt).

https://doi.org/10.1016/j.advengsoft.2022.103329
Received 27 July 2022; Received in revised form 17 September 2022; Accepted 19 October 2022
Available online 10 November 2022

the extended version is, lastly we had done a survey regarding the proposed estimation technique and done a comprehensive study with the existing estimation techniques but now we have applied the proposed estimation technique to projects and found outcomes [9, 10].

2. Significance of study

Due to the frequent change requests from a client in agile methodology enhances client satisfaction, but on other hand, it also increases the cost and time of project completion. The cause of the increase is because of agile's feature of change requests at any stage of the project/module. The main issue with this feature is that the product owner i.e. client controls the changes and requirements. Agile has different cost estimation techniques but all have some shortcomings. These shortcomings do not make such techniques pertinent for agile efforts estimation. Therefore, to overcome such shortcomings this study has proposed a cost estimation method with categorizations. We have done a survey study with the software teams and implemented the technique on real projects. This finding shows that the suggested method is more evaluable for effort prediction and calculates the more accurate cost and project completion time.

3. Related works

3.1. Existing techniques

The changes impact the project's achievement and fulfillment. In agile software development (ASD) the project manager welcomes the changes at any stage of the project [1, 13, 14]. The changes impact a task from numerous points of view because the changes have risks related to it as far as cost, time, and completion of the project. Then, over and over, change in demand for one, two, or more modules affect the entire project insufficiently because the Source Line of Code (SLOC) become increases which ultimately increases the project's size. As stated in the case study the maintenance cost has become high due to changes from the client. The study has examined the 2 studies as case samples and cost has become increasingly on the case sample which was more complex in code writing, cohesion in code, and changes in code modifications [38, 39, 42].

3.2. COCOMO I

The COCOMO I technique has been compared with the proposed technique. The reason to select/use the COCOMO I rather than others are:

The software industry's people which we target are still using and rating COCOMO I as compared to other estimation techniques. The reason for adoption is mentioned below that still researchers and practitioners are extending it.

The COCOMO is still in practice; researchers and practitioners are extending and modifying it with different concepts to introduce new effort estimation techniques as mentioned in these papers [14–17]. All the papers are published in IEEE publisher. There are also so many published articles based on COCOMO and published in reputed publishers [18–20].

Barry Boehm presented a model first time in 1981 called the COCOMO I model. The COCOMO represents the Constructive Cost Model [21–23]. In this condition, the KLOC/KDSL is speaking to the thousands of source lines of code.

Equations:

The equations of the COCOMO models are:

Effort=MM=a (KDSL/KLOC)b

Time=TDEV=2.5 × MMˆc

Software Cost = MM × per person salary per month.

Organic Mode

The natural improvement method of the COCOMO I model is little in size, less inventive, stable, and not a tight due date. It applies to the undertaking with the size of the source line of code being between 2-50 KLOC. It is the scope of the code of Organic mode [24, 25].

Semi-Detached Mode

The semi-segregated mode is the second method of improvement in the COCOMO I model. It is medium in size, inventive, due date and the advancement is normal. The scope of the size of the source line of code of this mode is between 50-300 KLOC [10, 26].

Embedded Mode

The Embedded Mode is the third method of the COCOMO I model. It is huge; profoundly inventive, tight due date, and the improved condition is unpredicTable The scope of the source line of code is >300 KLOC [27, 28].

3.3. Function point analysis

Software Change Effort Estimation (SCEE) is necessary not only during the process of software development as well as throughout the software maintenance process. Function Point Analysis (FPA), is used to calculate the amount of work required to estimate the software needs and wants of customers throughout the software maintenance cycle. Software archives are in a steady state during the maintenance stage [42]. Equation 1 illustrates how the FPA technique calculates the Function Points (FPs) of a product by merging Unadjusted Function Points (UFP) with Value Adjustment Factor VAF. UFPs are the total of all activities, including Internal Logical Files (ILFs), External Interface Files (EIFs), External Output Files (EOs), External Queries (EQs), and External Output Files (EIs) with their respective complexity levels (low, average and high). 14 General Unit Parameters can be used to determine the Value Adjustment Factor (VAF) (GSC) [12].

$$\text{FPs} = \text{UFP}^{*}\text{VAF} \tag{1}$$

3.4. Top-down or bottom-up estimation

Experts in software development effort assessment may use top-down or bottom-up approaches, i.e., they may base their assessments of the overall effort on the characteristics of the project in its entirety and disperse people across project tasks (top-down), or they could determine the overall effort prediction as the cumulative of the estimates for each project task (bottom-up). The study's emphasis on the expert estimating method was motivated by the prevalence of expert estimation, the absence of compelling information in support of formal estimation techniques, and the lack of knowledge of the characteristics of expert estimation procedures. The researchers looked at the top-down and bottom-up estimating procedures for the 2 categories of software development process expertise estimation strategies [15].

3.5. Price-to-win estimation

In this method, the client's budget greatly impacts how much the software project is estimated to cost. The cost of the project whatever the client is willing to spend on it, with the client's budget taking precedence over the software's functioning. This strategy is not advised since it places more emphasis on the client's capability and budget than it does on the functioning of the software. Low accuracy is assigned because accuracy differs greatly depending on the client's budget. It is not a useful method because it could delay development and deliveries and need the development team to put in extra hours. The client's budget and person-month ratio are used to validate this method [11].

3.6. Artificial neural network

Due to its propensity for arbitrary efficiency, neural networks are commonly employed as a technique for software effort predictions. Due to their learning capacity, artificial neural networks (ANN) have shown

Table 1
Motivational Techniques.

Techniques	Features adopted
Function Point	Low, Average, and High Categorizations
Expert Opinion	Professional Judgment About Project
Scrum Methodology	Daily Meeting Session
Delphi Technique	Prediction about Software

to be extremely effective in a wide range of real-world scenarios. For instance, ANN learning algorithms decide on weight alterations to enhance the cost function. The MRE across the present ANN output and the anticipated (target) outcome is the most effective but least frequently employed cost function because it can result in the lowest cost [17].

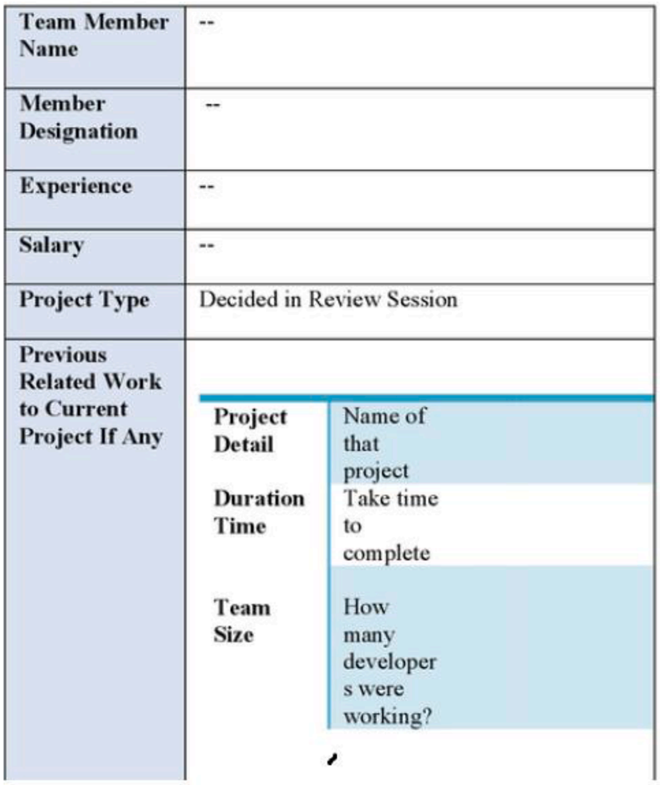
3.7. Support vector machine

Software effort estimation is the practice of calculating the amount of work necessary to create a software product (SEE). Support vector regression (SVR), a machine learning approach, is one of them that has been applied for cross-company (CC) large datasets effort prediction. Because SVR is capable of adapting to various and heterogeneous bits of data, it is useful [18].

This study has done a detailed literature analysis of existing works and the problem statement has supported the detailed related effort estimation techniques, some of these techniques help and motivate in proposing a new effort estimation technique. These techniques are Scrum methodology, function point, expert analysis, and Delphi technique as shown in Table 1. These all techniques have several limitations to measure the accurate cost and time for agile software development with change requests from the client [8–11]. Table 1 is describing the cost estimation techniques and adopted features for suggesting a new estimation technique.

3.8. Planning poker

Team members talk about cost and effort estimation using this method. Every team member participates in the conversation and shares their objectives for the estimation method because each participant has different criteria for estimation. Following a brief discussion, the team members compared each member's criteria to conclude the estimating procedure and requirements. To assure team member contact, utilize this strategy. This method is less useful in the software business and has



Team Member Name	--	
Member Designation	--	
Experience	--	
Salary	--	
Project Type	Decided in Review Session	
Previous Related Work to Current Project If Any	Project Detail	Name of that project
	Duration Time	Take time to complete
	Team Size	How many developer s were working?

Fig. 2. Module Info Finder

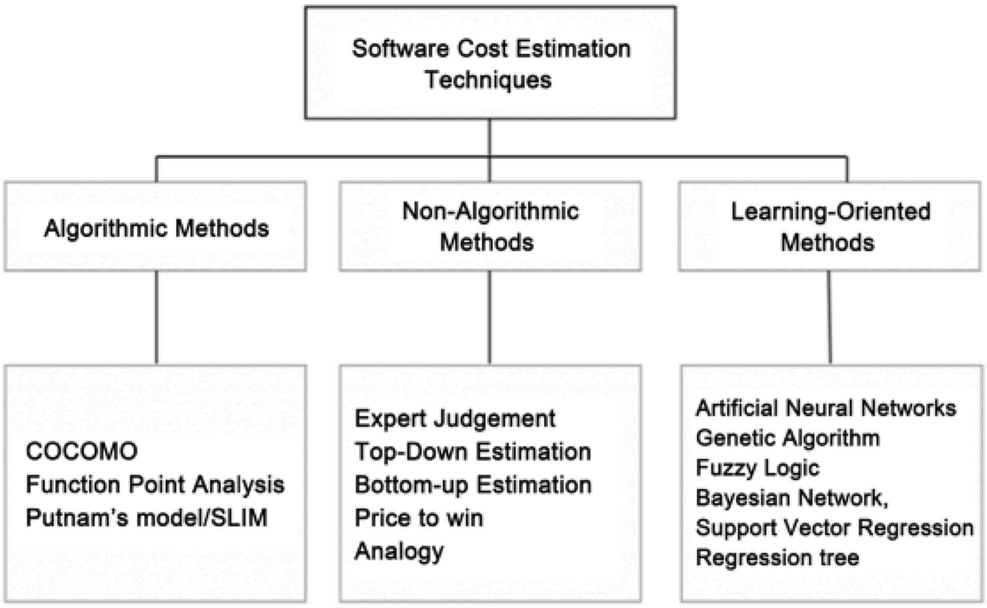


Fig. 1. Existing estimation techniques in software development [10].

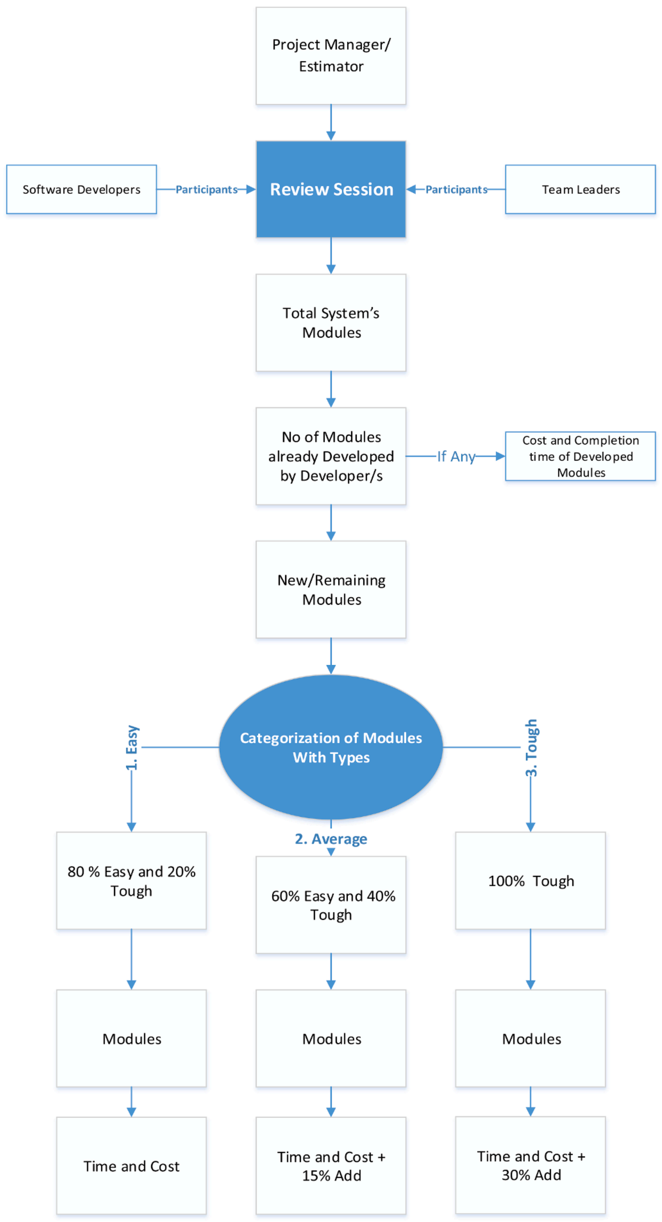


Fig. 3. Review Session Meeting [10].

less empirical support for accuracy. This method cannot be used to make predictions [32–35].

3.9. *Industrial surveys*

In this, we visited several software industries to fill out the questionnaire for the proposed technique. We have arranged meeting sessions and interviews with the developers, software architects, and project managers to get the outcomes of the project using the proposed estimation technique. This has revealed that the proposed estimation technique overcomes the issues of agile software development. The results of the meeting session got through the questionnaire (Questionnaire template as shown in the Appendix) [10,12].

4. Proposed model

We have proposed an estimation technique as a solution for software development to remove the issues. The proposed technique encourages the estimator to appraise the precise expense and time due to again and again changes that originate from the client. The technique is clarified through screens [7]. The author gave the name as Shariq Screen Model (SS Model) technique. Fig. 1

Screen1:

The estimation begins in the model during the survey session. The estimation session has appeared in Fig. 2. The effort estimator talks about the entire activities with the product engineers, and group pioneers and discovers the module's size, cost, timeframe, and exertion. The fundamental motivation behind the session is to share the development experience of software engineers, team managers with one another, and based on that experience, they can discover the efforts of the project. In the session's meeting, the individuals classify the modules into three kinds Easy, Average, and Complex modules of the software product. In modules (Easy, Average, and Complex) the project manager or estimator can choose the cost and time of the project. The session participants provide information regarding that how many modules have been developed before related to the present project and how many modules have not been worked on before similar to the current project. What no of the modules are new for the team? This survey session data helps the team to find the accurate time, effort, and cost in the survey meeting session [10, 12, 15].

Fig. 2 is giving the complete information of the review session meeting including team member names, designation, years of experience, salary, and project type has been decided in the review session means that this project falls under the categories of easy, complex, and tough. The "Previous Related Work" means a team member writes his/her previous completed project experience similar to the current project such as project name, completion time, and team size to complete that project. All information in the Module Info Finder helps the project manager to estimate the accurate project details.

4.1. *Module's categories*

4.1.1. *Easy*

The primary classification of the module is the simple class that implies the most extreme no of modules has been already developed in another project as like existing ones. The product design simply needs to replace every one of such modules with the present application or they need some minor improvements to meet current necessities as appeared in Fig. 3. The 80% simple and 20% extreme is the simple module class. In the model, 80% implies that 80% of work has been done as of now in other applications, and staying 20% is new for developers [10, 18, 25].

4.1.2. *Average*

The secondary classification is the Average module which implies normal quantities of modules have been developed in another project, the product design simply needs to replace these modules with the present application's modules or they need some changes as per the current project's requirements. In the normal class module, the quantity of recently developed modules is not exactly like simple module classification. As stated in Fig. 2 60% is simple and 40% is complex. In the model, 60% implies that 60% of work has been done on other project/s, and 40% is new for the team. The product developers need to put effort into this 40% part of the project [10, 18, 25].

4.1.3. *Difficult*

The third classification is the difficult modules which means the

Table 2
Presenting the working of screen 2.

Parameters	Example
Total modules of software	6
No of the completed module	2
SLOC of the completed module	20 KLOC
Remaining module	4
Total time completion of the completed module	4 months

entire project is new for developers. They have never developed any kind of such project earlier. They have no experience identified with the present project. The project is 100% difficult and new for the team. The developers need to spend a lot of effort and time to finish the projects and they need more resources to meet the requirements. The time, efforts, and cost of the project choose in the audit session [10, 18, 25].

All the categories and their values such as 20%-80%, 40%-60%, and 100% are decided after reviewing estimation techniques and discussion with developers and team leaders.

The total cost of the project can calculate by:

Total cost = Average Cost + Expected Cost

•Average cost

Here the average cost is representing the cost of previously developed modules in another application but is similar to a current project. It adds to the total cost of the current project after some changes and modifications to current project requirements [39–41].

•Expected cost

The expected cost is the cost decided in the review session by the participants in the session.

The total time of the project can calculate by

Total time= Average Time +Expected Time

•Average time

The average time is the completion time of the previously developed modules in another application but is similar to a current project. It adds to the total cost of the current project after some changes and modifications to current project requirements.

For Average Category project:

Total cost = Average Cost + Expected Cost + 15% ….1

Total time = Average Time + Expected Time +15% ….2

For Difficult Category project:

Total cost = Expected Cost + 30% …………3

Total time = Expected Time +30% …………4

In Eqs. 1–4 the 15% and 30% qualities are included by the project leader after the result of the Review Session. The objective of adding values is to remove the risk of project failure and complete the risks regarding time and effort increase. When the cost and time chosen in a session have been decided by the software developers then the project manager takes a total of 15% and 30% of the decided values to the definite cost and time secure tasks of the project.

Screen 2

This screen of the framework can help the estimator to predict the effort and size of the module to minimize the risk. Suppose when the change request comes after some modules have been developed then the project manager can estimate or predict the effort and size of the remaining modules with the change request. As shown in Fig. 3 modules 1, 2, and 3 have been developed with some source line of code, time, and cost without a change request but module 4 has a change request from the client, therefore, its size, effort, and impact can measure with the cost and size of already developed modules (1, 2 and 3) as shown in Fig. 3. After that, the estimator can predict the change request's time and cost from the already developed module's data. In this paper, we apply this screen in project 3 implementation. Given below are some parameters with an example for an estimator to estimate and predict the effort of the remaining modules as mentioned in Table 2.

Average of the size of the remaining modules:

$\sqrt[Completednoofmodule]{SLOC\ of\ completed\ module}$ = Average size of the remaining module.

In this phase, the project manager or meeting session members can calculate the size of the remaining modules of the project by the above-mentioned equations.

Screen 3

Screen 3 is managing the significant part when the venture advancement begins the group build up the primary module and sends it to the client for criticism and at the main module the change solicitation originates from the client then for the engineers and chief it is hard to appraise the size of current change solicitation and its effect on consummation time and cost of the rest of the modules in the task. In this

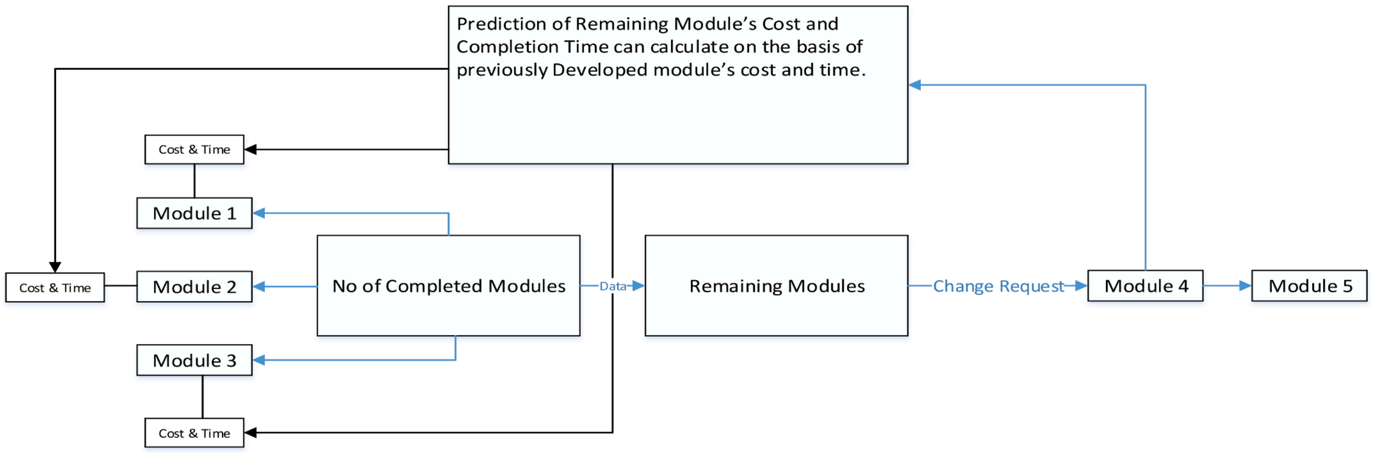

Prediction of Remaining Module's Cost and Completion Time can calculate on the basis of previously Developed module's cost and time.
Cost & Time
Module 1
Cost & Time
Module 2
No of Completed Modules
Data
Remaining Modules
Change Request
Module 4
Module 5
Module 3
Cost & Time

Fig. 4. Screen 1.

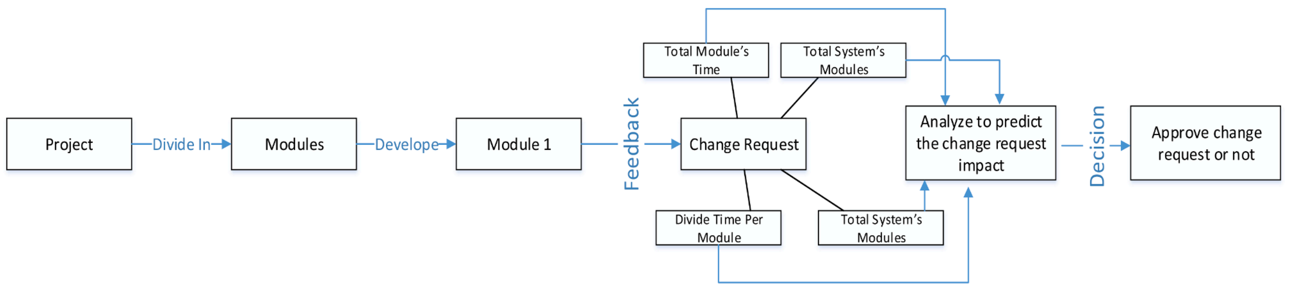


Fig. 5. Screen 2

Table 3
Presents the working of screen 3.

Parameters	Example
Total no of modules	5
Total time of the modules	12 months
Divide the time per module	2.4 months
Divide Effort per module	3 persons per month

Table 4
Adopted software industries.

Companies	Projects	No. Employees	Type of Services	Location	Sub-Locations
Company 1	1	200-250	Govt. software applications	Pakistan	Australia
Company 2	2	200-220	IT Solutions Consultancy	Pakistan	UK

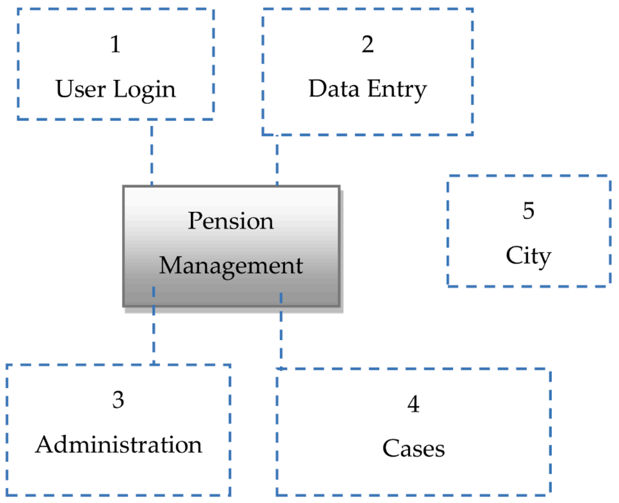


Fig. 6. Number of modules in project 1.

Table 5
Summary of Project 1.

Company	Company 1
Project Size	Medium
Project Nature	Pension management System
Team Size	6
KLOC	21
No. Modules	5
Completion Time	23 months
Already Developed Modules	2
Project Cost	12650 $

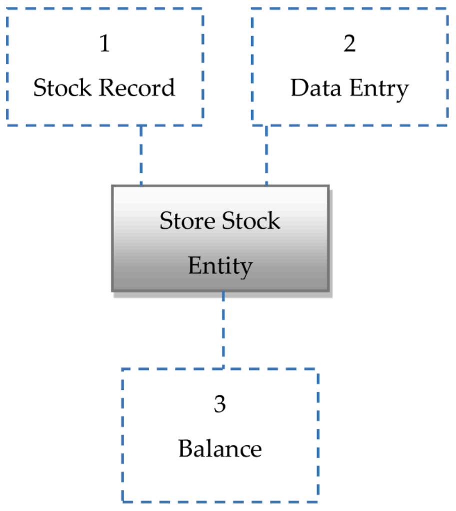


Fig. 7. Number of modules in project 2.

Table 6
Summary of Project 2.

Company	Company 2
Project Size	Medium
Project Nature	Store Stock Entry
Team Size	4
KLOC	15
No. Modules	3
Completion Time	15.6 months
Already Developed Modules	0
Project Cost	91000 $

way to deal with this, an estimator can utilize screen 3 to assess the size of the changes, the fulfillment time required, cost, and impact of progress changes in all tasks as appeared in Figs. 4 and 5. In Fig. 4 the project divides into modules, each module has some effort (Time and Cost) decided in the review session. Therefore when the change request comes at the first module then the estimator can predict its impact on other modules and also can analyze whether the change has to be fulfilled or not. After analyzing the estimator decides whether to approve the

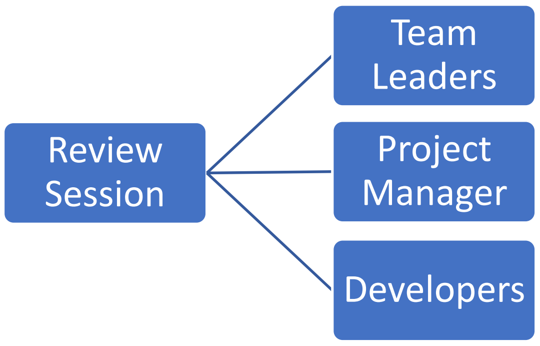


Fig. 8. Review session with the team for effort estimate.

change request or not. The estimator can prioritize the change requests as well. This screen helps the estimator to predict the effort for any change and its impact on the whole project. The parameters for this screen are shown with an example as mentioned in Table 3.

Screen 4

The screen is clarifying who is the suitable team member in the group to deal with the change request and difficult module of the project.

At the point when the venture begins then in the audit session the project's difficult module choose. After the revelation of these modules next errand is to discover a designer to work on these modules. Give every one of these modules to the most profoundly experienced designer in the group because of high experience and proficient code composing that make engineer novel from the different low experienced developers. The highly experienced developer/s deals with all change requests in the project [10, 29]. The selection of the highest experienced developer can be from the review session form mentioned in "Module Info Finder" the Manager/Estimator can also prioritize developers according to experience.

Fig. 3 explains the review session meeting, Here D is representing the Developer and T is representing Team Lead and P is representing the Project Manager, and Fig. 4 is explaining the overall picture of the proposed technique. Fig. 4 is explaining screen 2 and Fig. 5 is explaining screen 3.

4.1.4. *Implementation*

To evaluate the proposed technique different projects have been used. This technique has been applied to 2 software houses project data as stated in Table 4.

Project 1

Project 1 is a medium project type conducted for Company 1. The company is developing a Pension Management System application as described in Table 5.

Project 2

Project 2 is a large project type conducted for Company 2. The company is developing a Store Stock Entities application as described in Table 6.

Project 1

The project's 1 five modules are mentioned in Fig. 6:

1 User Login: The user can view his/her pensions and can get the update on upcoming pensions etc.
2 Data Entry: The data entry operator has to enter the data about all current and new pension holders.
3 Administration: Admin can view, update and delete any user History.
4 Cases: In the project, the phase of the case is related to the types of pension holders.
5 City Officer: The City officer can view all the data and pensions holders with a single click.

Project 2

The project's 2 modules are mentioned in Fig. 7:

1 Stock Record: The Stock Record keeps the record of entities in store.
2 Data Entry: The data entry operator enters the entity's data in stock.
3 Balance: The balance phase shows the total balance in the form of cash (sale $ purchase).

Most Change Requests on Project's Modules

In project 1 the change requests come from the client on modules are:

1 Administration: The admin facing that the updated data of some clients are not showing. Deletion and updating of data are not done.
2 User Login: In this module, the user is facing that some of his/her previous record or upcoming record is missed or someone else record is showing in their profile.
3 City Officer: The city officer when wants to see all cases of pensions then some cases are missed but entered by the data entry operator.

In project 2 the change requests come from the client on modules are:

1 Stock Record and Balance: The owner is facing that there is a difference between some entities' data in the stock record and the balance of entities is not match.

These are the reason these most change requests come from the client on the number of modules.

5. Results and analysis

Project 1

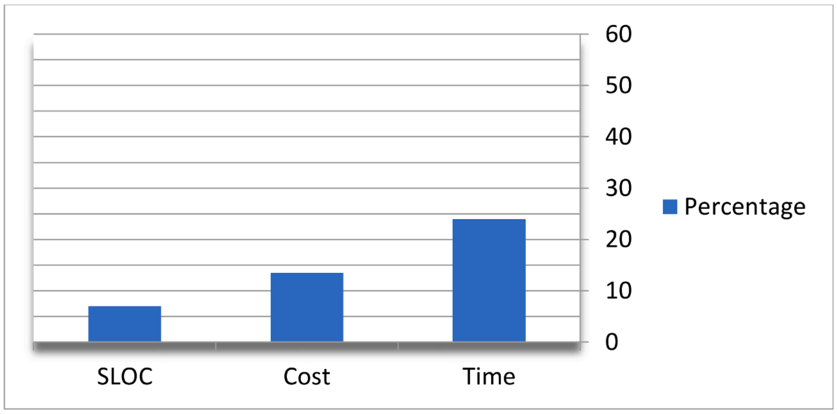


Fig. 9. Outcome of project.1 regarding changes.

Table 7
Outcome of development of Project 1.

No. Modules	3
Average SLOC	3000
Cost Required	1908 US $
Months for Completion	5.5

Table 8
Summary of Project.

	Actual	Final	Incremental
KLOC	21	24	3
Time	23 months	28.5 months	5.5 months
Cost	12650 $	14558 $	1908 $

Table 9
Postmortem Analyses of Techniques.

Features	*Techniques* COCOMO	SS Model
Project Suitable	Not Suitable for Agile	Suitable for Agile
Agile Software Development	X	√
Client Satisfaction	Less	High
Estimation Based On	SLOC	Review Session
Preliminary Data Required for estimation	√ Only Source Line of Code	√ Project's Detail i;e Project type, team size, etc
Modification	In the initial stage only when the project start	The runtime can do during project development
Knowledge Required	Product and Domain	Product and Domain
Extendibility	√	x
Project management involvement	x	√
Accuracy of Estimation	X	√

Project 1 is implemented in Company 1 on the pension management application. The proposed model is used to control the cost and time increment due to again and again change requests.

The cost and time of the project had decided after the review session in Company 1 as the proposed model was first to arrange the review session shown the Screen 1. In this project's review, session members were the project manager, 4 developers, and 3 team leaders with low and high development experience shown in **Fig. 8**.

The decided cost, completion time, effort, previously completed modules, SLOC, Team Size, etc. are mentioned in the Implementation Section. In the Implementation section that Project 1 has already developed 2 modules in some other projects similar to existing ones. The developers just have to make some minor changes to meet the current requirement and replace these 2 modules in the current project. Hence Project 1 falls under the Average Category by the proposed model's categories where the project is 60% easy and 40% tough according to the proposed model. Now the proposed model Screen 2 has been used here to estimate the cost, time, effort, and SLOC and filled the Module Info Finder template shown below.

Filled Review Session Meeting form for project 1.

Name	XYZ
Designation	Principle Architect
Experience	10 Years
Salary	150 K
Project Type	Average Category
Previous Related Work to Current Project If Any	

Project Detail	Employee Management
Duration Time	18 Months
Team Size	6

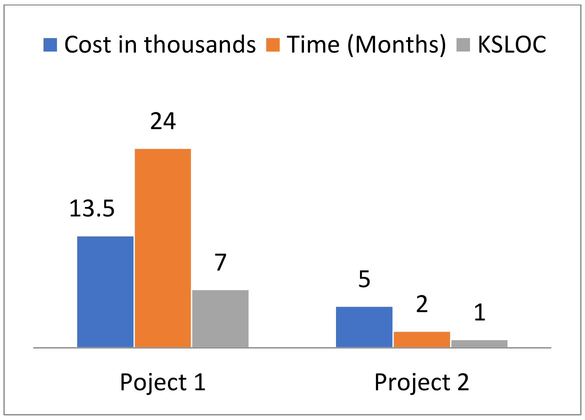


Fig. 10. Overall changes effects on projects.

Total modules of software	5
No of the completed module	2
SLOC of the completed module	8KLOC
Remaining module	3
Total time of completed module	5.5

Average of the 2 of the remaining modules: $\sqrt[Completednoofmodule]{SLOC\ of\ completed\ module} = \sqrt[2]{8\ KLOC} = 4$ KLOC per Module

Now the total time and cost can estimate by use of the Average cost and time estimation Formulas:

Total cost = Average Cost + Expected Cost + 15%.

Total cost = 3000 + 8000 + 1650 = 12650 US $

Total time = Average time + Expected time + 15%.

Total time = 5.5 + 14.5 + 3 = 23 Months

Then, change requests come from the client on 3 modules city officer, administration, and User Login as stated in Fig. 9.

The SLOC code of the whole project had become increased which ultimately increased the project's cost, effort, and completion time as stated in Table 7.

Hence the total project 1 values that are required to complete it with the change request come from the client as shown in Table 8:

Project 2

Project 2 is also implemented in Company 2. The developers have to develop the Stock Entity Application. The application is new for the developers and did not work before this time on such type of application. The application is new and tough for developers therefore after the review session the team decided on cost, effort, completion time, SLOC, etc. as mentioned in the project's description section. The project falls in the difficult category.

The formula from the proposed model used for this project is:

For Difficult Category project:

Total Cost = Expected Cost + 30%.

Total Cost = 7000 + 2100 = 9100.

Total time = Expected Time +30%.

Total time = 12 Months + 3.6 Months = 15.6 Months.

Here the Expected time and cost had been decided by the developers and the 30% accumulative value was added by the project manager to secure the project.

In project 2 most change requests that come from the client were on the Stock and Balance module. But the SLOC, Cost, and Time increased from the final values by the developers and project manager but the increment was very low and did not impact the project. The total analysis of projects 1, and 2 are shown in the graph. The Postmortem analysis graph is explaining the final result and analysis of all projects. Project 1 and 2 has some fault rate when the change request comes from

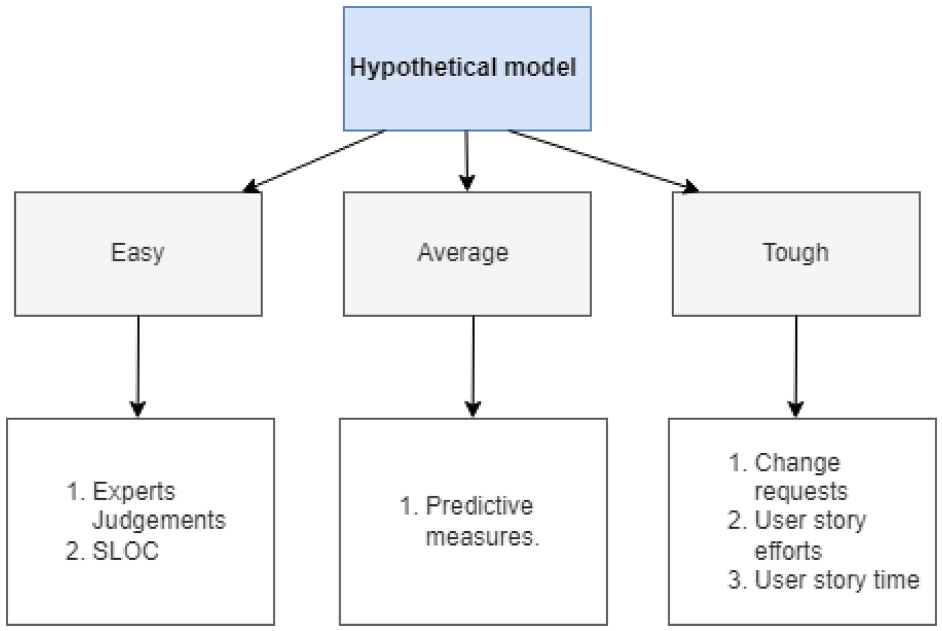


Fig. 11. Hypothetical model of variables.

Table 10
Specification of data, survey objectives, data sourcing, and data collection.

Objectives	To identify the main confronts and research objectives in agile cost estimation techniques.
Subject Area	Cost estimation
Main Research Question	What are the principal difficulties and research openings in cost estimation techniques?
More Specification Subject Area	Cost issues in agile
Interference	Surveys, definition, and other types of research related to agile research.
Type of Data	Questionnaires
How Data Was Acquired	Analyst develop Questionnaire for analysis
Data Format	Analyzed and statistical data
Experimental Factors	The data model consisted of software developers who mostly are software development persons.
Data Source Location	Software industries

Table 11
Surveyed Industries

Goal	Adoption of cost estimation technique in Industries.
Targeted Audience	Software Developers, Team Leaders, & Project Managers.
Data Collection Mode	Questionnaire & Interviews
No Industries for Survey	2
Total Questions	12
Participants	25

the client but the proposed model has less fault rate with the change request coming from the client rather than the COCOMO I model. Hence the COCOMO I model is not ingenious for agile software development. It needs some amendments to deal with the change request from the client side [42–44].

Postmortem Analysis of Cost, Time, SLOC in Projects is presented in Table 9 and Fig. 10.

6. Statically analysis

We have done a statistical analysis of the existing proposed estimation technique with the use of variables and survey sessions with the software industries.

Table 12
Hypotheses

HQ 1	Is the SS method helpful to manage the cost and time?
HQ 2	Is Review Session an efficient way to manage the project?
HQ 3	Can project cost and time be accurately estimated in the review session?
HQ 4	Do you think that the SS method can control the cost and time with the frequent change of requests?
HQ 5	Is the SS method easy to implement?
HQ 6	Do you think that the manager and client's relationship becomes enhanced through the SS method?
HQ 7	Can the SS method increase the company's business and reputation?
HQ 8	Are the SS method's all screens practical and easy to use?
HQ 9	Can the SS method's all screens control the cost and time to increase?
HQ 10	Is the SS method helpful for the project manager?
HQ 11	Is the SS method suitable for all SDLC models?
HQ 12	Can the SS method remove the cost and time issue from the Agile model?

Table 13
Variable alpha.

Dependent Variable	Cronbach's Alpha	Composite Reliability	(AVE)
Easy	.607	.725	.72
Average	.721	.818	.78
Tough	.604	.608	.56

6.1. *Variables and assessments*

In the research model, there are different types of variables that influence software development activities. It is a process influenced by six independent variables, three dependent variables, and one moderator variable. The questionnaire was used to determine six independent variables. Several questionnaires on a Likert scale rating are performed to assess their impact. The set of questions is intended to ask from the beginning of the questionnaire to examine the moderating variable software industries' size and environmental factors as the Fig. 11 is presenting the dependent and independent variables [30, 31]. We have used SPSS as a tool for data analysis and statistical results.

Table 14
Independent variables.

Predictive variables	Overall project success		Perceived Project Success		Organizational performance	
	Multiple R	R^2	B	SE	ß	t
Change request	0.67	0.38	0.41	0.48	0.46	0.27
The effort required for the user story	0.64	0.60	0.55	0.62	0.24	0.47
Predictive measures	0.72	0.50	0.34	0.35	0.45	0.64
Time required for user story	0.57	0.61	0.49	0.54	0.31	0.54
Developer's SLOC	0.50	0.52	0.60	0.60	0.20	0.28
Expert judgments	0.75	0.61	0.34	0.30	0.37	0.55

6.2. *Data gathering*

The questionnaires are designed by analyzing previous survey studies in the same field. Participants are asked a series of questions. Participants are asked broad questions related to the project estimation in the first phase of the survey. Assess the participant's designation to determine whether he or she is a manager or a developer. There are also some basic inquiries concerning their upcoming final product [32]. Table 10 is describing the data specification and survey objectives. Table 11 is presenting the survey industries and Table 12 is stating the hypotheses.

Because the questionnaire is slightly different from the previous one, a reliability test is required to ensure the questionnaire's accuracy. The degree to which a measurement model generates consistent and reliable outcomes is known as reliability. Cronbach's alpha coefficient is also called Alpha (α). It has many levels of sufficient outcomes. A dependability level of 0.7 is regarded as satisfactory. Cronbach's alpha is used for all independent variables. Most of the variables are greater than 0.5 as is shown in Table 13.

The validation of the hypothesis presented is shown in the Table 14. Among the independent and dependent variables, multiple bivariate tests are applied [34].

Multiple bivariate co-relation is used to test hypotheses from 1 to 6. The concept of bi-variant co-relation is used to examine the correlation between the two variables. It has a range of values from +1 to -1. +1 there is no relationship at zero. All of the factors in our study had a positive relationship. All of the variables are positively related to one another. And the independent variables are being used to examine their relationships. We conducted both zero-order and partial correlation analyses in Table 14 [35–37].

7. Conclusion

Agile has been invented to improve and overcome the deficiencies of efficient software development. At present, the agile model is used in software development vastly due to its support to both developers and clients resourcefully. COCOMO I is a famous cost estimation procedure however didn't ingenious for agile development when frequent change request comes from a client. It ultimately increases the cost and time of the project. This increase affects the budget and productive software development. There are different cost estimation techniques in software development but all have deficiencies that make them less efficient. Due to such reasons, this study has proposed a cost estimation technique. The proposed estimation technique is predictions-based and has different categorizations of projects based on user stories complexities and the developer's expertise. We have applied the suggested technique to ongoing projects to find the results and effectiveness. We have used two projects with different sizes and user stories. Both projects have different modules and developers with different expertise. The outcomes of the project development stated that the proposed technique is valuable more than the existing techniques because it categorized projects on basis of the developer's experience.

Agile development needs an increasingly precise and numerical model to appraise the cost, time, exertion, and culmination time of the task. The product advancement with enormous scale ventures and with other SDLC models like the cascade, and winding likewise required estimation Systems to help these sorts of development.

Author's contribution

All authors are contributed equally.

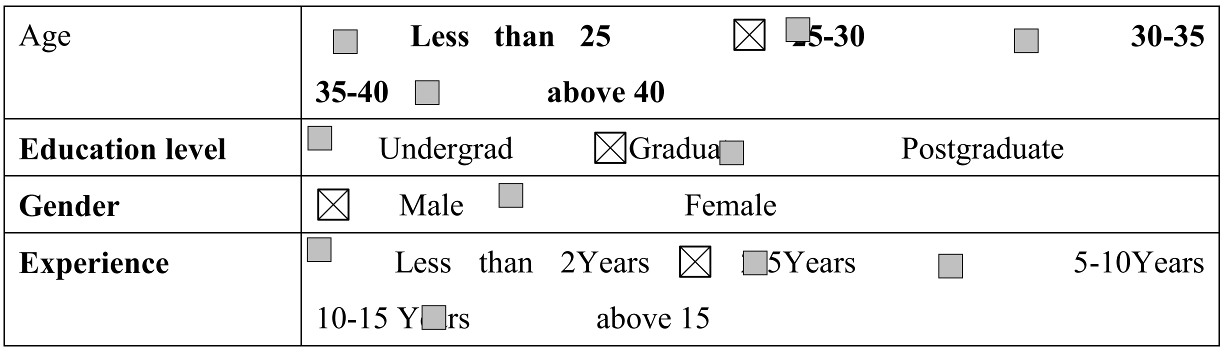
Declaration of Competing Interest

The authors declare that they have no Conflict of interest.

Data Availability

No data was used for the research described in the article.

Appendix



Age	☐ Less than 25 ☒ ☐5-30 ☐ 30-35 35-40 ☐ above 40
Education level	☐ Undergrad ☒Gradua☐ Postgraduate
Gender	☒ Male ☐ Female
Experience	☐ Less than 2Years ☒ ☐5Years ☐ 5-10Years 10-15 Y☐rs above 15

(continued on next page)

(*continued*)

11	Is the SS method suitable for the all SDLC models?	☐	☐	☒	☐	☐
12	Can the SS method remove the cost and time issue from the Agile model?	☐	☐	☐	☐	☒

11	Is the SS method suitable for the all SDLC models?	☐	☐	☒	☐	☐
12	Can the SS method remove the cost and time issue from the Agile model?	☐	☐	☐	☐	☒

References

[1] Kuhrmann M, Tell P, Hebig R, Klunder JAC, Munch J, Linssen O, Richardson I. What makes agile software development agile. IEEE Trans Software Eng 2021.

[2] Khalid A, Butt SA, Jamal T, Gochhait S. Agile scrum issues at large-scale distributed projects: scrum project development at large. Int J Softw Innov (IJSI) 2020;8(2): 85–94.

[3] Kaim R, Härting RC, Reichstein C. Benefits of agile project management in an environment of increasing complexity—a transaction cost analysis. Intelligent decision technologies 2019. Singapore: Springer; 2019. p. 195–204.

[4] Khmelevsky Y, Li X, Madnick S. Software development using agile and scrum in distributed teams. In: 2017 Annual IEEE International Systems Conference (SysCon). IEEE; 2017, April. p. 1–4.

[5] Rodríguez P, Haghighatkhah A, Lwakatare LE, Teppola S, Suomalainen T, Eskeli J, Oivo M. Continuous deployment of software intensive products and services: a systematic mapping study. J Syst Softw 2017;123:263–91.

[6] Rasheed A, Zafar B, Shehryar T, Aslam NA, Sajid M, Ali N, Khalid S. Requirement engineering challenges in agile software development. Math Probl Eng 2021:2021.

[7] Ramessur MA, Nagowah SD. A predictive model to estimate effort in a sprint using machine learning techniques. Int J Inform Technol 2021;13(3):1101–10.

[8] Obilor NA, Chibuike AAOAB, Donatus NO. Constructive cost model II metrics for estimating cost of indigenous software. Int J Adv Eng Res Sci 2021;8:7.

[9] Butt SA, Jamal T. Frequent change request from user to handle cost on project in agile model. Proc Asia Pacific J Multidiscipl Res 2017;5(2):26–42.

[10] Chirra SMR, Reza H. A survey on software cost estimation techniques. J Softw Eng Applic 2019;12(06):226.

[11] Shah J, Kama N. Extending function point analysis effort estimation method for software development phase. In: Proceedings of the 2018 7th International Conference on Software and Computer Applications; 2018, February. p. 77–81.

[12] Venkatesh V, Thong JY, Chan FK, Hoehle H, Spohrer K. How agile software development methods reduce work exhaustion: Insights on role perceptions and organizational skills. Inform Syst J 2020;30(4):733–61.

[13] Kaur I, Narula GS, Wason R, Jain V, Baliyan A. Neuro fuzzy—COCOMO II model for software cost estimation. Int J Inform Technol 2018;10(2):181–7.

[14] Jørgensen M. Top-down and bottom-up expert estimation of software development effort. Inform Softw Technol 2004;46(1):3–16.

[15] Jorgensen M. Relationships between project size, agile practices, and successful software development: results and analysis. IEEE Softw 2019;36(2):39–43.

[16] Rankovic N, Rankovic D, Ivanovic M, Lazic L. A new approach to software effort estimation using different artificial neural network architectures and Taguchi orthogonal arrays. IEEE Access 2021;9:26926–36.

[17] Villalobos-Arias L, Quesada-López C, Guevara-Coto J, Martínez A, Jenkins M. Evaluating hyper-parameter tuning using random search in support vector machines for software effort estimation. In: Proceedings of the 16th ACM International Conference on Predictive Models and Data Analytics in Software Engineering; 2020, November. p. 31–40.

[18] Rygge H, Jøsang A. Threat poker: Solving security and privacy threats in agile software development. In: Nordic Conference on Secure IT Systems. Cham: Springer; 2018, November. p. 468–83.

[19] Nhung HLTK, Hoc HT, Hai VV. A review of use case-based development effort estimation methods in the system development context. In: Proceedings of the Computational Methods in Systems and Software; 2019. p. 484–99.

[20] Shekhar S, Kumar U. Review of various software cost estimation techniques. Int J Comput Applic 2016;141(11):31–4.

[21] Dalal S, Dahiya N, Jaglan V. Efficient tuning of COCOMO model cost drivers through generalized reduced gradient (GRG) nonlinear optimization with best-fit analysis. Progress in Advanced Computing and Intelligent Engineering. Singapore: Springer; 2018. p. 347–54.

[22] Kumar KH, Srinivas K. Preliminary performance study of a brief review on machine learning techniques for analogy based software effort estimation. J Amb Intell Human Comput 2021:1–25.

[23] Aizaz F, Janjua UI, Zafar H, Khan JA, Kazim I. An empirical investigation on software cost estimation techniques and barriers on agile software development in software industry of Pakistan. In: 2021 International Conference on Frontiers of Information Technology (FIT). IEEE; 2021, December. p. 194–9.

[24] Sánchez R, Pinto-Roa DP. A new approach to software effort estimation using linear genetic programming. Proceed Ser Braz Soc Comput Appl Math 2018;6(1).

[25] Sakhrawi Z, Sellami A, Bouassida N. Support vector regression for enhancement effort prediction of Scrum projects from COSMIC functional size. Innov Syst Softw Eng 2022;18(1):137–53.

[26] Kaushik A, Tayal DK, Yadav K. The role of neural networks and metaheuristics in agile software development effort estimation. Research Anthology on Artificial Neural Network Applications. IGI Global; 2022. p. 306–28.

[27] Boehm BW. Software cost estimation meets software diversity. In: Software Engineering Companion (ICSE-C), 2017 IEEE/ACM 39th International Conference on. IEEE; 2017, May. p. 495–6.

[28] Przybylek A, Zakrzewski M. Adopting collaborative games into agile requirements engineering. In: 13th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE`18); 2018. https://doi.org/10.5220/0006681900540064.

[29] Zhang Z. The benefits and challenges of planning poker in software development: comparison between theory and practice (doctoral dissertation. Auckland University of Technology; 2017.

[30] Przybylek A, Albecka M, Springer O, Kowalski W. Game-based Sprint retrospectives: multiple action research. Empir Softw Eng 2022;27. https://doi.org/10.1007/s10664-021-10043-z.

[31] Khuat TT, Le MH. A novel hybrid abc-pso algorithm for effort estimation of software projects using agile methodologies. J Intell Syst 2018;27(3):489–506.

[32] Menzies T, Yang Y, Mathew G, Boehm B, Hihn J. Negative results for software effort estimation. Empir Softw Eng 2017;22(5):2658–83.

[33] Hamid M, Zeshan F, Ahmad A, Ahmad F, Hamza MA, Khan ZA, Aljuaid H. An intelligent recommender and decision support system (IRDSS) for effective management of software projects. IEEE Access 2020;8:140752–66.

[34] Lee DH, Chang IH, Pham H, Song KY. A software reliability model considering the syntax error in uncertainty environment, optimal release time, and sensitivity analysis. Appl Sci 2018;8(9):1483.

[35] Silhavy R, Silhavy P, Prokopova Z. Using actors and use cases for software size estimation. Electronics 2021;10(5):592.

[36] Khan SA, Alenezi M, Agrawal A, Kumar R, Khan RA. Evaluating performance of software durability through an integrated fuzzy-based symmetrical method of ANP and TOPSIS. Symmetry 2020;12(4):493.

[37] Lee DH, Chang IH, Pham H. Software reliability model with dependent failures and SPRT. Mathematics 2020;8(8):1366.

[38] Plebankiewicz E, Meszek W, Zima K, Wieczorek D. Probabilistic and fuzzy approaches for estimating the life cycle costs of buildings under conditions of exposure to risk. Sustainability 2019;12(1):226.

[39] Sahu K, Alzahrani FA, Srivastava RK, Kumar R. Hesitant fuzzy sets based symmetrical model of decision-making for estimating the durability of Web application. Symmetry 2020;12(11):1770.

[40] Sjøberg DI, Anda B, Mockus A. Questioning software maintenance metrics: a comparative case study. In: Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE; 2012, September. p. 107–10.

[41] Arora M, Sharma A, Katoch S, Malviya M, Chopra S. A state of the art regressor model`s comparison for effort estimation of agile software. In: 2021 2nd

International Conference on Intelligent Engineering and Management (ICIEM). IEEE; 2021, April. p. 211–5.

[42] Butt SA, Khalid A, Ercan T, Ariza-Colpas PP, Melisa AC, Piñeres-Espitia G, Ortega RM. A software-based cost estimation technique in scrum using a developer's expertise. Adv Eng Softw 2022;171:103159.

[43] Blinowski G, Ojdowska A, Przybyłek A. Monolithic vs. microservice architecture: a performance and scalability evaluation. IEEE Access 2022;10:20357–74.

[44] Butt SA, Misra S, Luis DMJ, Emiro DLHF. Efficient approaches to agile cost estimation in software industries: a project-based case study. In: International Conference on Information and Communication Technology and Applications. Cham: Springer; 2020, November. p. 645–59.