Received: February 15, 2018 205

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

An Enhanced Framework for Effort Estimation of Agile Projects

Atef Tayh Raslan[1]* **Nagy Ramadan Darwish[2]**

[1]*Department of Computer Science, Institute of Statistical Studies and Research, Cairo University, Egypt*
[2]*Department of Information Systems and Technology, Institute of Statistical Studies and Research, Cairo University, Egypt*
* Corresponding author's Email: atef_rslan2004@yahoo.com

Abstract: Software effort estimation methods are used to measure the total time and forecast the amount of required effort to develop the software. In addition, agile software projects, the requirements are characterized by changeability during the software projects. Therefore using the traditional estimation models in agile software projects may cause inaccurate effort estimation. This paper proposes a framework which utilizes The Constructive Cost Model (COCOMO II), story points, and fuzzy logic models without affecting the sacredness of agile principles. Furthermore, the results show the proposed framework increasing the value of Prediction Level (Pred) to 80%.

Keywords: Agile software development, Effort estimation, COCOMO II, Story point, Fuzzy logic.

1. Introduction

Software project management is the process of planning, organizing, staffing, monitoring, controlling and leading a software project [1]. The characteristics of agile software development process include: modularity on development process level, iterative with short cycles, time-bound with iteration cycles, economic in development process, adaptive, incremental, minimize the risks, people oriented, and collaborative and communicative [2]. Large scale can be defined by the amount of time devoted to the project; the amount of people working on the project, and how many lines of code the software has [3]. The agile methods are difficult to scale up to larger projects due to the lack of sufficient architecture planning, but they emphasize the benefits of agile development when the future requirements are highly unpredictable [4].

Large-scale agile involves additional concerns in interfacing with other organizational units, such as human resources, marketing and sales, and product management [5]. There are many challenges facing the large-scale agile projects; challenges in regard to realize continuous testing, increased maintenance effort with an increase in the number of releases, management overhead due to the need for coordination between teams, detailed dependencies are not discovered on a detailed level due to lack of focus on design, long requirements engineering duration, due to complex decision processes in requirements engineering, requirements priority lists are hard to create and maintain, waiting times in the process, specifically in design waiting for requirements, reduction of test coverage due to shortage of projects and lack of independent testing, and increased configuration management effort according to some researchers existing agile principles that do not support distributed development environment architecture [6].

This paper aims to introduce an approach for estimating the agile project in different scales. The proposed framework is based on mixing between the story point's method and COCOMO II method. Also, the framework utilizes the fuzzy inference system to increase the estimation accuracy.

The remainder of this paper is organized as follows. Section 2 describe the agile software development. The COCOMO II method and story point's method are discussed in Section 3. Section 4 describes the related works in effort estimation. Section 5 provides the architecture and process of

the proposed framework. Section 6 introduces the experiential analysis. Finally, the conclusions and the future work are described in section 7.

2. Agile software development

Agile software development is an iterative and incremental approach that is performed in a cooperative manner to produce high quality software that meets the changing requirements of the users [7]. Agile software development methods are particularly designed to deal with change and uncertainty [3]. There are many agile methods including eXtreme Programming (XP), Scrum, Agile Model Driven Development (AMDD), Dynamic System Development Methodology (DSDM), Feature-Driven Development (FDD), Adaptive Software development (ASD), and Lean Software Development (LSD) [8]. These methods promote development, teamwork, collaboration, and process adaptability throughout the life cycle of the project [9].

In this paper we will describe the AMDD as an example to illustrate the agile software development lifecycle. AMDD also helps to scale agile software development when the team is large and/or distributed and when "the team" is the entire IT effort at the enterprise level [10]. Fig. 1 shows the AMDD lifecycle for a software project which includes four activities: Envisioning, Iteration modelling, model storming, and Test-driven development (TDD).

Model storming is just in time (JIT) modelling: you identify an issue which you need to resolve, you quickly grab a few team mates who can help you, the group explores the issue, and then everyone continues as before [10]. The project team uses a modelling tools (such as whiteboard) and explore the problem until solve it.

Test-Driven Development (TDD) is a technique for building software that guides software development by writing tests [11]. TDD is a technique that concerns the development process, rather than the structure of the developed product [12].

The agile principles were written by agile manifesto authors and their value which are based on individuals and interactions at all stages of the project, customer collaboration at each end of the iteration and responding to change based on the customer requirements has to be reflected at any stage[13].

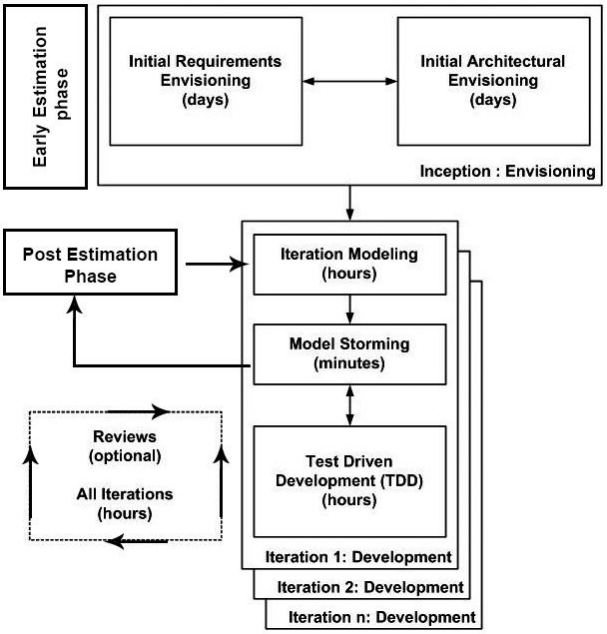

Figure. 1 The AMDD lifecycle

3. Effort estimation techniques

Software cost estimation is the set of techniques and procedures that organizations use to arrive at an estimate for proposal bidding, project planning and probability estimates [14]. The most common estimation techniques; COCOMO II Method and story point(s) method. These methods are described in details in the following subsections.

3.1 Story point's method

The story points refer to measuring unit to estimate the size of story in agile developments. Story points are usually expressed by sequence of numbers where each number is the sum of the previous two, this method called by Fibonacci series method [1, 15]. The story point's estimation approach is processed in the following order: Story size estimation, complexity estimation, implementation level estimation, velocity estimation, velocity optimization, and overall project effort. The teamwork estimates stories by using a relative scale to each story. The most common methods can be used for story size estimation is a planning poker method [15]. The complexity refers to a measure of the resources expended by a system while interacting with a piece of software to perform a given task [16]. Also, the teamwork uses the Fibonacci series method to scale the complexity. Velocity is the volume of work accomplished in a specified period of time, by a given team [17]. The velocity in agile refers to amount of stories that can be handled in single iteration. Velocity is a measure

of a team's rate of progress. It is calculated by adding the number of story points assigned to each user story that the team completed during the iteration.

Implementation Level Factors (ILF) includes all factors that reflect the level of understanding the project components by the team members. Also, the level of implementation uses the Fibonacci series method to scaling the each factor.

Velocity optimization phase refers to studying the project constraints which should be completed before the calibration to improve the stability of the velocity calculation. This process includes two factors; Friction factors (FR) and Dynamic force factor (DF). The friction includes a range of factors that may affect the team velocity which are team composition, process, environmental factors, and team dynamic.

The story point's estimation formulas showed in Eqs. (1), (2), and (3):

$$User\ Story\ Effort = \sum_{1}^{N} Complexity \times ILF \times Story\ Size \quad (1)$$

Where;
N: Total number of user story

$$Velocity = Initial\ project\ velocity\ (DF \times FR) \quad (2)$$

$$Project\ Time = \frac{User\ Story\ Effort}{Project\ Velocity \times Worked\ days/month} \quad (3)$$

3.2 The COCOMO II method

The Constructive Cost Model (COCOMO II) method was developed upon the COCOMO-81 model, which provides two main models early design model and post architecture model. The Early Design model of the COCOMO II is used for estimating the cost and effort values of an incomplete project and product analysis roughly. COCOMO II includes five different Scaling Factors (SFs); Precedentedness (PREC), Process Flexibility (FLEX), Risk Resolution (RESL), Team Cohesion (TEAM), and Process Maturity (PMAT). Each of these factors is rated in 6 levels ranging between "Very Low" to "Extra High".

This model includes three sub modules: Applications composition, early design and post architecture. The application composition model is used to estimate effort and schedule on projects that use integrated computer aided software engineering tools for rapid application development [18].

The early design model includes the estimation of the project after collecting the requirements. The early design stage includes studying all design alternatives based on function points, five scales factors, and seven effort multipliers. The post architecture model focuses on the project details after project's overall architecture is developed. Eq. (4) shows the effort estimation in Person/Month (PM):

$$PM = A \times (S)^{E} \times \prod_{i=1}^{n} EM_i \quad (4)$$

Where;
A: Constant based on the type of the project (Organic, semidetached, and embedded).
S: Refer to the software size which expressed in a Kilo Line of Code (KLOC).
EM: Effort multiplier.
E: Constant used to estimate the development effort which presented in Eq. (5).

$$E = B + 0.01 + \sum_{i}^{5} SF_i \quad (5)$$

B: Constant based on the juvenility of the software, development flexibility, risk management methods and the process maturity).
SF: Scale factor weights (very low, low, nominal, high, very high, and extra high).

Moreover, the COCOMO II is an algorithmic model which provides a set of tools and techniques for evaluating the effects of software technology improvements on software life cycle costs and schedules [19]. It can be used with a story point method to estimate the effort in a large scale projects.

Further, the story point and COCOMO models do not support the imprecision and uncertainty associated with the effort estimation attributes such story size, scale factors, complexity, and velocity.

3.3 Fuzzy logic

Fuzzy Logic (FL) is a methodology to solve problems which are too complex to be understood quantitatively. It is based on fuzzy set theory and introduced in 1965 by Lotfy Zadeh [20]. Fuzzy logic provides the concept of fuzzy sets to handle vague and inaccurate data [1]. The Fuzzy Logic System deals with fuzzy parameters, which address imprecision and uncertainties, by mapping out the path of a given input to an output using the computing framework called the Fuzzy Inference System (FIS).

There are many membership functions but in this research, the triangle membership function will be used which is represented Eq. (6) [1]:

$$Triangle(x:a,b,c)=max(min(\frac{x-a}{b-a},\frac{c-x}{c-b}),0) \quad (6)$$

A fuzzy inference engine is a collection of IF - THEN rules stored in fuzzy rule base is known as inference engine. Defuzzification is the process which refers to the transform of fuzzy output into crisp output. The most common tool used for fuzzy systems is MATLAB which used for defining the input, output, inference rules, and the shape of membership function for the fuzzy system [1].

4. Related works

There are many researches in effort estimation and agile software development. Some of these researches provide good ideas in our work. The following are some examples of these literatures:

Abeer H. introduced model that aims to increasing the sensitivity of COCOMO cost model. This model uses a fuzzy model to enhances the accuracy and sensitivity of COCOMO 81 intermediate by using a fuzzifying the cost drivers. The researcher contributed to the increasing the sensitivity of COCOMO81 cost model [21].

Abeer H. Brought model better estimation version primarily based a genetic fuzzy system. This model makes use of a genetic algorithm with COCOMO81 intermediate. The consequences confirmed that the accuracy was improved compared with traditional COCOMO model [22].

Carl Friedrich Kreß et al introduced scaling agile estimation methods with a parametric cost model. This research presents three solutions based on story points and COCOMO II to improve the estimation method for large agile projects [23].

Sathish Kumar C., et al, introduced guide based on harmony search algorithm to optimize the effort estimation on agile software development. The results show the proposed plan gives an affecting estimation compared with other estimation methods [13].

Assem H. Mohammed and Nagy Ramadan Darwish introduced a proposed fuzzy based framework for calculating success metrics of agile software projects. The main idea in this research is calculating the Success metric value (SMV) based on the Success factors values (SFV) and the Importance value (IM) for each success factor value (SF). The proposed framework enables the agile stakeholders to represent the values of the success factors in a human-like language [24].

Wilson and Corinne introduced a set of practical effort estimation models for software development projects during the contract bidding phase. The study is based on data collected from 196 previous projects from the United States Department of Defense delivered from 2005 to 2016. Also, the authors claimed that models may be used for agile projects. Moreover, this study concluded that the accuracy improved when peak staff and supper domain are added as inputs to the calculation [25].

Simon introduced the proposed a regression model to predict the effort required to design small and medium scale software projects. Also, this study used 60 previously developed software projects. On my opinion, this study was focused on single software company, so the study results may not generalize to a real environment [26].

Based on this literature, there are studies focused on improving the COCOMO model by using the fuzzy model [21-22]. Also, some of the researchers trying to improve the effort estimation in the agile environment by using the story points, COCOMO II, fuzzy logic, harmony algorithm, and regression method [13, 23-24, 26]. Moreover, there is a study discussed the effort estimation during the bidding phase in agile methods [25].

These studies don't cover the agile environment characteristics. So, we introduce the proposed framework using the fuzzy logic method to increase the accuracy of effort estimation in agile methods.

5. The proposed framework

The agile development life cycle includes four phases; inception, construction phase, transition phase, and production phase. The inception phase is a first phase in the life cycle which includes the requirements envisioning and planning the initial project resources. The construction phase focuses on the development and testing for each project unit. Furthermore, the construction phase could be implemented by different agile methods. The transition phase includes many tasks such finalizing testing, finalizing documentation, users training, and running pilot programs. The production phase aims to keep the project useful and more efficient after it has been deployed.

The proposed framework starts with the envisioning phase which performed during the first week of a project, as shown in Fig. 2. The envisioning phase aims to identify the scope and architecture of the project accordance with the available requirements. Moreover, it provides a preliminary estimation for the project schedule and

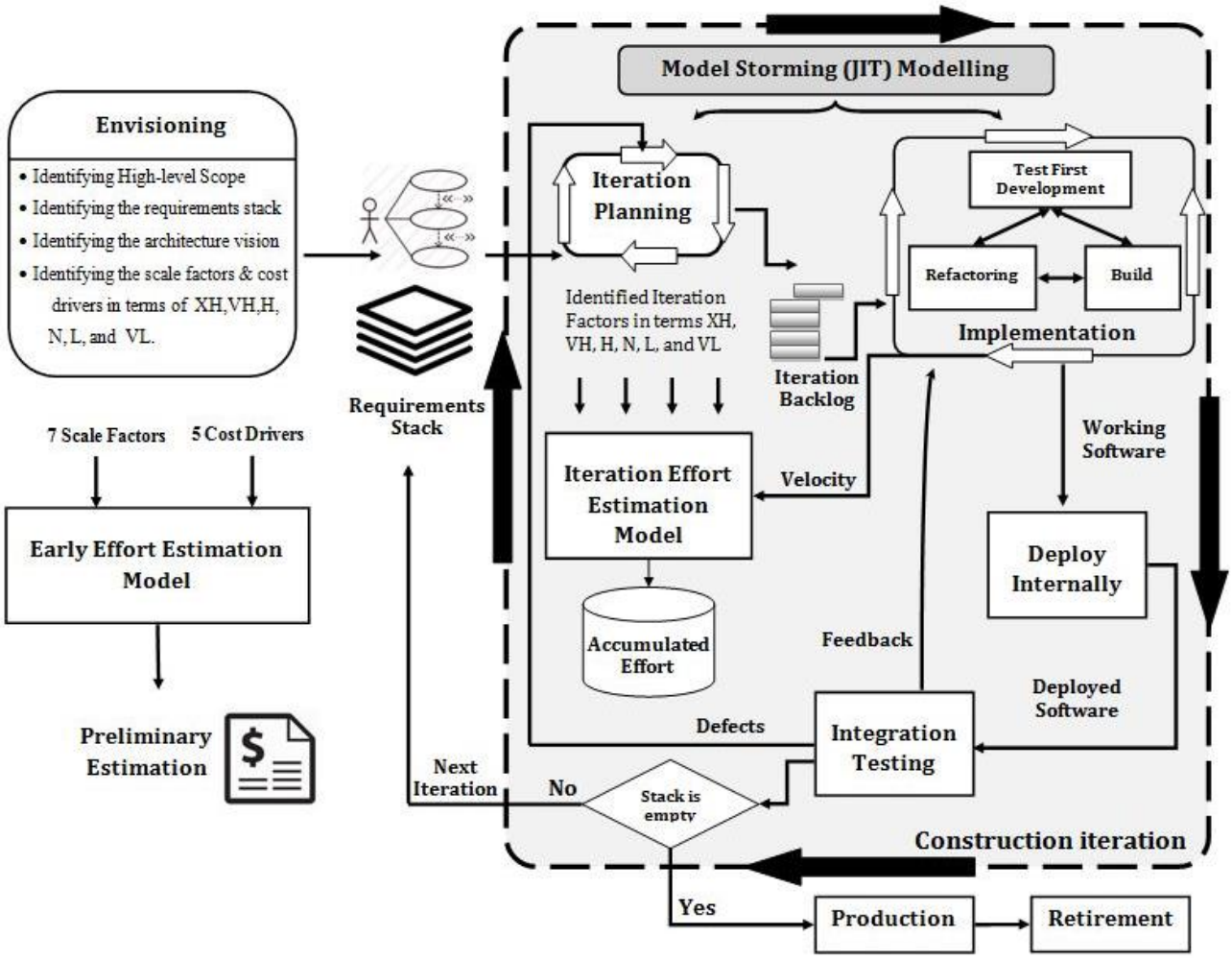

Figure. 2 The proposed framework

budgets without an extravagant cost of documentation.

The Early Effort Estimation Model (EEEM) is a fuzzy logic based which helps the project team to make early effort judgments. The EEEM uses a COCOM early design factors classified as 5 Scale factors and 7 cost drivers. The scale factors include Precedentedness (PREC), Process Flexibility (FLEX), Risk Resolution (RESL), Team Cohesion (TEAM), and Process Maturity (PMAT).

Table 1 shows the COCOMO scale factors and their weights ranked from very low to very high. The PREC reflects the similarity between the current project and historical developed projects. The FLEX reflects the amount of the elasticity in the development process.

The RESL factor represents the amount of risk that may have faced the project during the development life cycle. The TEAM factor reflects the extent to which the project's team members know each other and worked well together previously.

The PMAT reflects the amount of the strength ripeness model of the organization that is realizing the project. Furthermore, the COCOMO cost drivers includes 7 factors includes the Product Reliability and Complexity (RCPX), Developed for Reusability (RUSE), Platform Difficulty (PDIF), Personnel Capability (PERS), Personnel Experience (PREX), Schedule (SCED), and Support Facilities (FCIL). These factors rated in 6 levels ranging between "very low" to "extra high" [27].

Fig. 3 shows the EEEM which accepts 5 scale factors, 7 cost drivers, and project size in Kilo Line of Code (KLOC) as inputs and produces the project preliminary estimation. Each of these factors has a numerical value named scale factor weight.

All input variables in EEEM mutated to the fuzzy sets based on the fuzzification process. The terms Very Low (VL), Low (L), Nominal (N), High (H), Very High (VH), and Extra High (EH) were defined for the 12 variables, early design properties and scale factors, in COCOMO II.

Table 1. Scale factors

SCALE FACTOR	SYMBOL	VL	L	N	H	VH
PREC	SF1	6.20	4.96	3.72	2.48	1.24
FLEX	SF2	5.07	4.05	3.04	2.03	1.01
RESL	SF3	7.07	5.65	4.24	2.83	1.41
TEAM	SF4	5.48	4.38	3.29	2.19	1.10
PMAT	SF5	7.80	6.24	4.68	3.12	1.56

Table 2.COCOMO II cost drivers

COST DRIVERS	EL	VL	L	N	H	VH	EH
RCPX	0.73	0.81	0.89	1.0	1.30	1.74	2.38
RUSE	-	-	0.95	1.0	1.07	1.15	1.24
PDIF	-	-	0.87	1.0	1.29	1.81	2.61
PERS	2.12	1.62	1.26	1.0	0.83	0.63	0.50
PREX	1.59	1.33	1.12	1.0	0.87	0.71	0.62
SCED	1.43	1.30	1.10	1.0	0.87	0.73	0.62
FCIL	0.00	1.43	1.14	1.0	1.0	1.0	-

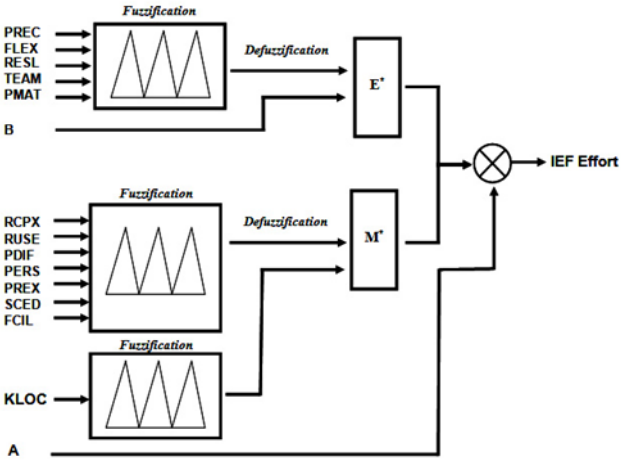

Figure. 3 The early effort estimation model (EEEM)

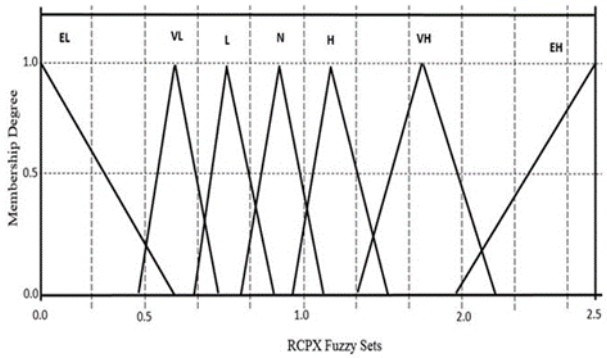

Figure. 4 Antecedents fuzzy sets of RCPX factor

The KLOC variable represents the project size which transformed to fuzzy sets in terms of Small, Medium, Large, and Extra Large. The software size categorization based on the definition is defined as a large system software project is about 10,000 function points, greater, or about 128 KLOC, while a super large system was taken to be 512 KLOC or more [28].

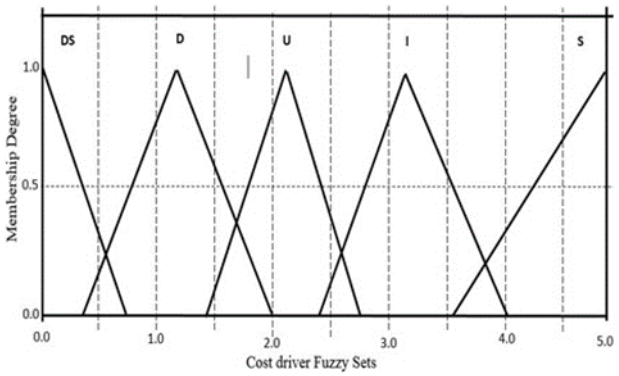

Figure. 5 Consequent cost driver of RCPX factor

Table 2 shows the COCOMO II cost drivers which are ranked in seven levels Extra Low (EL), Very Low (VL), Low (L), Nominal (N), High (H), Very High (VH), and Extra High (EH).

For example, in case of Product Reliability and Complexity (RCPX) cost driver, we define a fuzzy set for each linguistic value with a Triangular Membership Function (TRIMF).

Fig. 4 shows the fuzzy sets of the antecedent part which are derived using the definition of the RCPX levels (EL, VL, L, N, H, VL, and EH) where are given by Table 2. All inputs converted to fuzzy sets by using a triangular membership function, as shown in Eq. (6).

The effort selection in EEEM upon on 5 inputs factors and 5 membership functions ranked as Increases Significantly (IS), Increased (I), Unchanged (U), Decreased (D), and Decreased Significantly, where the inputs factors and membership functions are produced 55 rules. The consequent of ETC factor showing in Fig. 4.

The fuzzy sets of the consequent part that are derived using the RCPX factor values given by Table 2, as shown in Fig. 5.

The user requirements in agile development methods are subject to changes during the development phase. Developing in iterations allows the development team to adapt quickly to changing requirements [29]. After all requirements identified in the EEEM the project team creating a stack of user requirements which are ranked by their priority.

The Iteration Effort Estimation Model (IEEM) is an iterative model that starts after the project architecture was defined in EEEM. Fig. 6 shows the IEEM which classified into a two sub-models as velocity factors and COCOMO factors. The velocity sub-model accepts two inputs Friction factors (FR) and Dynamic force factor (DF).

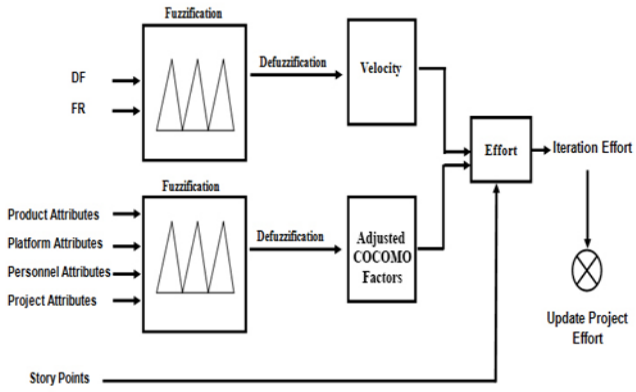

Figure. 6 The iteration effort estimation model (IEEM)

The FR and DF factors in velocity sub-model converted to fuzzy variables based on the fuzzification process. The DF consists of 9 factors could affect the velocity, these factors ranked as Normal (N), High (H), Very High (VH), and Extra High (XH), as shown in Table 2. The FR includes 4 forces could slow down the project development process. Each of FR forces scaled according to amount of the risk as Stable(S), Volatile (V), Highly Volatile (HV), and Extra Volatile (XV), as shown in Table 4.

In COCOMO sub-model, the estimated effort for the development of the agile project is calculated using 19 factors and output adjusted cost drivers. The 19 input variables Represented in Impact of Software Failure (FAIL), Product Complexity (CPLX), Developed for Reusability (RUSE), Required Software Security (SECU),Platform Constraints (PLAT),Platform Volatility (PVOL), Analyst Capability (ACAP),Programmer Capability (PCAP),Personnel Continuity (PCON), Applications Experience (APEX), Language and Tool Experience (LTEX), Platform Experience (PLEX), Precedentedness (PREC), Development Flexibility (FLEX), Opportunity and Risk Resolution (RESL), Stakeholder Team Cohesion (TEAM), Process Capability & Usage (PCUS),Use of Software Tools (TOOL), and Multisite Development (SITE).

Table 3 shows the COCOMO II factors which ranked as Very Low (VL), Low (L), Nominal (N), High (H), Very High (VH), and Extra High (XH).

In the fuzzification phase, we have defined fuzzy groups corresponding to the various associated linguistic values for each attribute. The triangular membership function is used to define the linguistic values for each factor based on its definition. The model rule base contains the linguistic variables related to the agile project. Each rule uses a connective number of rules which have used in this model is more than 300 rules for the velocity and cost drivers.

Table 3. COCOMO II factors

Attributes	Cost Drivers	VL	L	N	H	VH	XH
Product factors	FAIL	0.82	0.92	1.00	1.10	1.26	
	CPLX	0.73	0.87	1.00	1.17	1.34	1.74
	RUSE		0.95	1.00	1.07	1.15	1.24
	SECU	0.90	0.94	1.00	1.19	1.36	1.88
Platform factors	PLAT			1.00	1.11	1.29	1.63
	PVOL		0.87	1.00	1.15	1.30	
Personnel factors	ACAP	1.42	1.19	1.00	0.85	0.71	
	PCAP	1.34	1.15	1.00	0.88	0.76	
	PCON	1.29	1.12	1.00	0.90	0.81	
	APEX	1.22	1.10	1.00	0.88	0.81	
	PLEX	1.19	1.09	1.00	0.91	0.85	
	LTEX	1.20	1.09	1.00	0.91	0.84	
Project factors	TOOL	1.17	1.09	1.00	0.90	0.78	
	SITE	1.22	1.09	1.00	0.93	0.86	0.80
	PREC	6.20	4.96	3.72	2.48	1.24	
	FLEX	5.07	4.05	3.04	2.03	1.01	
	RESL	7.07	5.65	4.24	2.83	1.41	
	TEAM	5.48	4.38	3.29	2.19	1.10	
	PCUS	7.80	6.24	4.68	3.12	1.56	

The defuzzification is the process of converting from fuzzy sets to crisp sets. In this research, we use the MATLAB tool for calculation. Furthermore, the crisp output is calculated by using the Center of Area (COA) method, as shown in Eq. (7) [1].

$$COA=\frac{\sum_{x=a}^{B}\mu(X).X}{\sum_{x=a}^{B}\mu(X)} \quad (7)$$

In the proposed model, we use a triangular membership function in formula (6) to obtain the fuzzy set. The Adjusted COCOMO Factors (ACF) calculated using formula (8). The $\mu_{A^{(x)_i}}$ is the membership function of the fuzzy set Ai associated with the cost driver xi. Eq. (9) shows the calculation of velocity (V) using a triangular membership function of fuzzy set associated with FR and DF factors.

$$ACF = \sum_{I=1}^{19} \mu_{A^{(x)_i}} \; . \; EM_i \quad (8)$$

$$V = (\prod_{i}^{4} \mu_{A^{(x)_i}} \; . \; FR_i).(\prod_{j}^{9} \mu_{A^{(y)_j}} \; . \; DF_j) \quad (9)$$

Square series has been proved to be the most preferred series in agile estimation since it provides realistic level of accuracy for complex and will-defined project [30]. Consequently, we use a square series (1, 4, and 9) for calculating the story point's intensity levels. The next step is calculating the Iteration Story Points (ISP) through Eq. (10).

$$\text{ISP} = \sum_{I=1}^{number\ of\ stories} SP_i + (0.1 \times ACF) \quad (10)$$

Table 4. Velocity factors

Friction Factors (FR)	S	V	HV	XV
Team composition	1	.98	.95	.91
Process	1	.98	.94	.89
Environmental factors	1	.99	.98	.96
Team dynamic	1	.98	.91	.85
Dynamic Factors (DF)	N	H	VH	XH
Expected to team change	1	.98	.95	.91
Introduction to a new tools	1	.99	.97	.96
Vendor's defect	1	.98	.94	.90
Team member's responsibilities outside the project	1	.99	.98	.98
Personal issues	1	.99	.99	.98
Expected delay in stakeholder response	1	.99	.98	.96
Expected ambiguity in details	1	.98	.97	.95
Expected changes in environment	1	.99	.98	.97
Expected relocation	1	.99	.99	.98

In order to estimate actual project time (PT) based on the amount of story points and agile team velocity, as showing in Eq. (11).

$$T = \frac{ISP}{V} \times \frac{1}{working\ days\ per\ month} \quad (11)$$

6. Experimental analysis

In this section, a sample of dataset which extracted from COCOMONASA2 dataset will be used. It was collected from six NASA centers and covers a wide range of software domains, development process, languages and complexity, as well as fundamental differences in culture and business practices between each center [21].

The study includes 10 projects, each project has a cost drives, scale factors, project size in KLOC, and estimated effort using EEEM model. The results evaluated via Magnitude of Relative Error (MRE) and Prediction Level (PRED) metrics. Eq. (12) shows the MRE which is used to measure the error contained in the estimated value regardless of whether the error is positive or negative [28].

$$MRE_i = \frac{|Actual\ Effort_i - Predicted\ Effort_i|}{Actual\ Effort} \quad (12)$$

The PRED metric is the prediction at level *L*, as shown in Eq. (13):

$$PRED\ (L) = \frac{K}{N} \times 100 \quad (13)$$

Where, *k* is the number of observations where MRE is less than or equal to *L*, and *N* is the total number of observation [31].

Table 5 shows the sample of ISP for stories were collected during the project construction under agile methodology. Table 6 shows the iteration results for each project. The accumulated effort was calculated during the construction phase. For example, the first project was constructed during five iterations, the preliminary effort estimation is 104.97, the accumulated effort is 85.82 and the MER is 0.22.

The MMER represents the average of MER for all projects in dataset. Finally, the values of MER, MEER, and PRED are calculated using the COCOMONASA dataset, as shown in Table 6. As a result, the MMER and PRED become 0.25 and 80.0% respectively. Moreover, the data set in this study was calculated using the COCOMO II method, story point method, and the proposed framework.

Table 5. ISP results

Story(#)	SP	ACF	0.1 * ACF	ISP
1	34	115.20	11.52	45.52
2	13	48.11	4.81	17.81
3	8	127.50	12.75	20.75
4	5	128.90	12.89	17.89
5	8	122.00	12.20	20.20
6	21	114.33	11.43	32.43
7	8	98.18	9.82	17.82
8	8	121.23	12.12	20.12
9	21	127.16	12.72	33.72
10	34	135.13	13.51	47.51
Total				273.77
Working days/month				22.00
Velocity				0.87
Actual Effort				14.30

Table 6. Project iterations results

Project (#)	Output from EEEM	Output from IEEM	MER
1	104.97	85.82	0.22
2	99.4	75.91	0.31
3	29.69	36.27	0.18
4	31.7	27.29	0.16
5	37.75	33.56	0.12
6	8.06	7.65	0.05
7	13.06	11.06	0.18
8	280.63	147.37	0.9
9	24.82	22.72	0.09
10	36.8	29.68	0.24
		MMER	0.25
		PRD(0.25)	80.00%

7. Conclusion and future work

In this research a framework based on story points and COCOMO cost drivers have been proposed. These factors converted to fuzzy sets based on their definition. The proposed model consists of two phase's preliminary and constructive iteration phase. The early effort estimation phase (EEEM) focuses on estimation before developing phase to help the agile team to makes effort judgments.

The IEEM model uses a fuzzy logic which accepts five cost drivers, seven cost drivers, and size of requirements in LOC. The Constructive iteration phase calculates the effort based on iteration velocity, the intensity levels using SP, and COCOMO post design attributes. In addition, all inputs converted to fuzzy sets using a triangular membership function.

The rule base contains a set of conditional statements that define the cost drivers, scaling factors, LOC, DF, FR, product attributes, platform attributes, personal attributes, and project attributes. Consequently, the fuzzy sets are defuzzfied to crisp values using COA method and then the effort is estimated in each phase. The use of fuzzy logic in the proposed model may increase the effort estimation.

The researchers thought that the utilization of story point with COCOMO factors may reduce the risk of falling project in chaos by providing realistic effort in constructive iteration phase. Moreover, the use of the proposed model increases the value of PRED from 70% to 80%. As a result, the accuracy of effort estimation improved.

The ideas that are expected to be focused in the future include:

- Using different membership function(s) and comparing between the produced efforts.
- The proposed model can also be extended utilizing Neuro-Fuzzy method.
- Improving the proposed framework by utilizing a training algorithm like genetic algorithms (GA) to tune the fuzzy sets parameters.

References

[1] A. Raslan, N. Darwish, and H. Hefny, "Towards a Fuzzy based Framework for Effort Estimation in Agile Software Development", *International Journal of Computer Science and Information Security*, Vol.13, No.1, pp. 37-45, 2015.

[2] J. Hunt, *Agile software construction*, Springer, ISBN-10: 1-85233-944-6, 2006.

[3] R. Morken, *Coordination in Large-Scale Agile Development*, Master of Science in Informatics, Norwegian University of Science and Technology, 2014.

[4] H. Saeeda, F. Arif, N. Minhas, and M. Humayun, "Agile Scalability for Large Scale Projects: Lessons Learned", *Journal of Software*, Vol.10, No. 7, pp. 893-904, 2015.

[5] K. Dikert, M. Paasivaaraa, and C. Lassenius, "Challenges and success factors for large-scale agile transformations: A systematic literature review", *The Journal of Systems and Software*, Vol. 119, pp. 87–108, 2016.

[6] K. Petersen and C. Wohlin, "A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case", *Journal of Systems and Software*, Vol. 82, No. 9, pp.1479-1490, 2009.

[7] N. Darwish, "Towards an Approach for Evaluating the Implementation of Extreme Programming Practices", *International Journal of Computer Science and Information Security*, Vol. 9, No. 11, pp. 37-45, 2011.

[8] T. Chow and D. Cao, "A survey study of critical success factors in agile software projects", *Journal of Systems and Software*, Vol. 81, pp. 961– 971, 2008.

[9] Z. Ani and S. Basri, "A case study of effort estimation in agile software development using use case points", *Sci. Int. (Lahore)*, Vol. 5, No. 4, pp.1111-15, 2013.

[10] S. Ambler, "Agile Software Development at Scale", In: *Proc. of Second IFIP TC 2 Central and East European Conference on Software Engineering Techniques*, 2007.

[11] M. Fowler, "Test Driven Development", http://martinfowler.com/bliki/TestDrivenDevelopment.html , visited on 2018.

[12] T. Verhoeff, "Programming Methods (2IP15): Test-Driven Development", https://www.win.tue.nl/~wstomv/edu/2ip15/downloads/Series_07/slides_14.pdf, visited on 2018.

[13] C. Kumar, A. Kumari, and R. Perumal, "An Optimized Agile Estimation Plan Using Harmony Search Algorithm", *International Journal of Engineering and Technology*, Vol. 6, No. 5, pp. 1994-2001, 2014.

[14] S. Bhandari and P. Kakkar, "Soft computing based technique for accurate effort estimation: A survey", *International Journal of Engineering Sciences*, Vol. 9, pp.14-22, 2013.

[15] M. Cohn , *Agile Estimating and Planning*, 1st Edition, Prentice Hall, Pearson Education, ISBN-13: 978-0131479418, 2006.

[16] P. Fitsilis and V. Damasiotis, "Software Project's Complexity Measurement: A Case Study", *Journal of Software Engineering and Applications*, Vol. 8, pp. 549-556, 2015.
[17] W. Hayes, S. Miller, M. Lapham, E. Wrubel, and T. Chick, *Agile Metrics: Progress Monitoring of Agile Contractors*, Software Engineering Institute, Mellon University, 2014.
[18] B. Boehm, C. Abts, and S. Chulani, "Software development cost estimation approaches –A survey", In: *Proc. of Annals of Software Engineering*, Vol. 10, pp. 177–205, 2000.
[19] S. Shekhar and U. Kumar, "Review of Various Software Cost Estimation Techniques", *International Journal of Computer Applications*, Vol. 141, No. 11, pp.31-34, 2016.
[20] M. Ganesh, *Introduction to Fuzzy Sets and Fuzzy Logic*, Prentice-Hall, 2006.
[21] A. Hamdy, "Fuzzy Logic for Enhancing the Sensitivity of COCOMO Cost Model", *Journal of Emerging Trends in Computing and Information Sciences*, Vol.3, No.9, pp. 1292-1297, 2012.
[22] A. Hamdy, "Genetic Fuzzy System for Enhancing Software Estimation Models", *International Journal of Modeling and Optimization*, Vol. 4, No. 3, pp. 227-232, 2014.
[23] C. Kreß, O. Hummel, and M. Huq, "Scaling Agile Estimation Methods with a Parametric Cost Model", In: *Proc. of the Ninth International Conference on Software Engineering Advances*, 2014.
[24] A. Mohammed, and N. Darwish, "A Proposed Fuzzy based Framework for Calculating Success Metrics of Agile Software Projects", *International Journal of Computer Applications*, Vol. 137, No. 8, pp.17-22, 2016.
[25] W. Rosa and C. Wallshein, *Software Effort Estimation Models for Contract Cost Proposal Evaluation*, ICEAA Professional Development & Training Workshop, 2017.
[26] S. Kuan, "Factors on Software Effort Estimation", *International Journal of Software Engineering & Applications*, Vol.8, No.1, pp.23-32, 2017.
[27] A. Kinra, "A Fuzzy Based Model for Software Quality Estimation Using Risk Parameter Assessment", *International Journal of Computer Science and Mobile Computing*, Vol. 3, No. 3, pp.807 – 814, 2014.
[28] E. Manalif, *Fuzzy Expert-cocomo Rrisk Assessment and Effort Contingency Model in Software Project Management*, Master of engineering science, University of Western Ontario London, Ontario, Canada, 2013.
[29] D. Cohen, M Lindvall, and P. Costa, "An Introduction to Agile Methods", *Advances in Computers*, Elsevier, Vol. 62, 2004.
[30] S. Bhalerao and M. Ingle, "Incorporating Vital Factors in Agile Estimation Through Algorithmic Method", *International Journal of Computer Science and Applications*, Vol. 6, No. 1, pp. 85 – 97, 2009.
[31] I. Attarzadeh and S. Ow, "Soft Computing Approach for Software Cost Estimation", *International Journal. Of Software Engineering*, Vol. 3, No. 1, pp. 3-12, 2010.