A NOVEL EFFORT ESTIMATION MODEL FOR SOFTWARE REQUIREMENT CHANGES DURING SOFTWARE DEVELOPMENT PHASE

Jalal Shah, Nazri Kama and Nur Azaliah A Bakar

Department of Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia

ABSTRACT

Software Requirements Changes is a typical phenomenon in any software development project. Restricting incoming changes might cause user dissatisfaction and allowing too many changes might cause delay in project delivery. Moreover, the acceptance or rejection of the change requests become challenging for software project managers when these changes are occurred in Software Development Phase. Where in Software Development Phase software artifacts are not in consistent state such as: some of the class artifacts are Fully Developed, some are Half Developed, some are Major Developed, some are Minor Developed and some are Not Developed yet. However, software effort estimation and change impact analysis are the two most common techniques which might help software project managers in accepting or rejecting change requests during Software Development Phase. The aim of this research is to develop a new software change effort estimation model which helps software project manager in estimating the effort for software Requirement Changes during Software Development Phase. Thus, this research has analyzed the existing effort estimation models and change impact analysis techniques for Softwrae Development Phase from the literature and proposed a new software change effort estimation model by combining change impact analysis technique with effort estimation model. Later, the new proposed model has been evaluated by selecting four small size software projects as case selections in applying experimental approch. The experiment results show that the overall Mean Magnitude Relative Error value produced by the new proposed model is under 25%. Hence it is concluded that the new proposed model is applicable in estimating the amount of effort for requirement changes during SDP.

KEYWORDS

Software Change Effort Estimation, Software Requirement Changes, Function Point Analysis, Constructive Cost Model and Software Development Phase.

1. INTRODUCTION

Software Project Management (SPM) has existed for years, but it still remains a great challenge for software project team to produce successful software that fulfill its end user requirements within the predicted time and cost [1]. Several studies have highlighted the importance of the role of software project manager in project's success or failure. Moreover, Software project manager plays a vital role in a software development team and after all he is responsible for the success or failure of the project [2]. According to, Lehtinen, et al. [3] a software project failure means an

identifiable failure in the cost, schedule, scope, or quality of the project. In addition, Kaur and Sengupta [4] stated that the most common reasons for project failure are rooted in the project

management process itself and, they have identified some estimation mistakes in their research which are: unclear project goals, objectives, and project requirement changes during the project. Therefore, a software project manager is always responsible in managing the Software Requirement Changes (SRCs) and justifies the decisions that he has taken while accepting or rejecting SRCs [5].

SRCs may occur at any phase of Software Development Life Cycle (SDLC)[6]. Accommodating a huge amount of SRCs might increase the development time and cost of the software and denying many software requirements changes possibly increase customer dissatisfaction[7]. However, a good change acceptance decision can help software project manager in managing SRCs [8]. Kama and Halmi [9], [10] stated that there are two most related inputs that help software project manager in an effective change acceptance decision for SRCs during SDP are: (i) Change Impact Analysis (CIA) and (ii) Software Change Effort Estimation (SCEE) [11]. CIA is the process of predicting the impact of SRCs on software artifacts and it also identifies the factors that need to be modified to accomplish a Software Requirement Change (SRC). Alternatively, SEE is a process that predicts the amount of work that is required to implement a software requirement change [9, 12].

There are two types of effort estimation models which are widely used: (i) algorithmic-based models and (ii) non-algorithmic-based models. Some of the most common algorithmic-based models are: COCOMO II [13], Function Point Analysis [14] and Use-Case Points [15]. Whereas, some of the non-algorithmic-based models are: Expert Judgement [16], Analogy Based Estimation [17] and Delphi [18]. Although, several extensions of these models are developed to estimate effort in Software Development Phase (SDP). However, there are not many studies to estimate effort in SDP and still it remains an interesting task for software project managers to estimate the amount of effort for SRCs in SDP [11, 19-21].

Although, [9] stated that the combination of CIA and SEE may improve the estimation accuracy for SDP. At present, few studies [9, 11, 22] have been identified which used the combination of CIA and SEE as an effort estimation model and provided better estimation results for SDP [9, 23].

This research presents a new Software Change Effort Estimation Model (SCEEM) that can be used in measuring the amount of effort for SRCs during SDP. The new model identifies and considers the related factors that contribute to the effort estimation for SRCs in SDP.

This paper is structured as follows: Section (2) presents related work, section (3) describes proposed model, section (4) presents evaluation process and section (5) presents conclusion and future work.

2. RELATED WORK

The five most related keywords involved in this research are Software Change Effort Estimation, Change Impact Analysis, Software Development Phase, Function Point Analysis and Constructive Cost Model II.

2.1. SOFTWARE CHANGE EFFORT ESTIMATION

Software Change Effort Estimation (SCEE) is the process of predicting that how much work and how many hours of work are required to develop a software. Normally it describes in man-days or man-hours unit [24]. Several SCEE models have been developed such as: Expert Judgement [19, 25]; Estimation by Analogy [26]; Source Lines of Code [27] Function Point Analysis [28] and Regression Analysis [19]. These models are dived in two categories: non-algorithmic based software change effort estimation models and algorithmic based software change effort estimation models.

Non-algorithmic models rely on learning, understanding and analyzing previous software projects and may include past personal experiences. Two most common non-algorithmic SCEE models are: Expert Judgement and Estimation by Analogy. Expert Judgement is a non-algorithmic SEE model [17]. According to Sufyan, et al. [29] usually software development teams preferred to use expert judgement effort estimation model instead of other estimation models because of its flexibility and simplicity. Furthermore, they mentioned that it is not sure whether the estimation results which are produced during this model are hundred percent accurate or not. Additionally, it is also observed that unstructured expert judgment is extremely unpredictable. According to Idri, et al. [26] and Sufyan, et al. [29], estimation by analogy predicts the amount of effort required for new software projects. Whereas, for prediction of required effort it uses the statistics of the similar software projects which have been developed earlier. Furthermore, they have stated that because of simplicity and flexibility of analogy model it is used frequently as a hybrid model. Whereas, in hybrid model, two or more SCEE models are combined for the reason to improve the accuracy and performance such as particle swarm optimization (PSO), grey relational analysis (GRA), artificial neural network (ANN), principle component analysis (PCA), and rough set theory [30].

While, Algorithmic models are constructed based on fixed and predefined statistical and mathematical equations. Some of the most common algorithmic-based SCEE models are: Source Lines of Code, Function Point Analysis and Regression Analysis. Source Lines of Code (SLOC) is the most ordinarily used metric to denote size of software during SEE in algorithmic methods [27]. On the other hand, Hira, et al. [13] stated that estimating with SLOC is nearly impossible until the complete development. Furthermore, they have stated that SLOC gives two different estimation results for different programming languages, because the number of lines of code in each language are different. Function Point Analysis (FPA) is also an algorithmic-based SCEE model [31]. It is used for measuring the size and complexity of a software by calculating the functionality, that system provided to its user [32]. While, in late 1970s Allan Albrecht acknowledged that measuring effort estimation by SLOC is insufficient. For that reason, in 1979 he introduced FPA method [33].

Regression analysis is an algorithmic-based SCEE model and it has another way of calculating effort estimation [34]. This model uses mathematical approaches for measuring effort estimation. Furthermore, it uses two variables such as: (i) SLOC and (ii) FPA for software size measurement. On the other hand, some regression analysis models use different parameters such as: operating system or programming language for an independent variables [35]. The benefit of using regression analysis model for effort estimation is its accuracy in measurement. Among other regression analysis models the most famous one is Constructive Cost Model (COCOMO) II

presented by Boehm [36]. Some earlier researchers have stated the significance of this method for estimating effort [13, 37].

2.2. CHANGE IMPACT ANALYSIS

Change impact analysis (CIA) is the process of identifying potential consequences of a change, or estimating what needs to be modified to accomplish a change [12]. The motivation behind CIA research is to identify the implications of change on software artifacts. The change can be in any form such as addition, modification or removal of existing or new software artifacts. Whereas, the information of affected artifacts can help software project managers in taking effective decisions regarding the change. As, Asl and Kama [38] stated that change impact analysis can helps project managers in decisions making, that whether to accept or reject the change based on predicted consequences.

Several CIA techniques have been developed such as: Use Case Maps (UCM) technique [39], Class Interactions Prediction with Impact Prediction Filters (CIP-IPF) technique [38], Path Impact technique [24], Influence Mechanism technique [37], and SDP Change Impact Analysis Framework (SDP-CIAF) [40]. These techniques are divided into two categories named as: Static Impact Analysis and Dynamic Impact Analysis. Static Impact Analysis (SIA) technique considers static information from software artifacts to produce a set of possible impact classes. Some of the common SIA techniques are: Use Case Maps (UCM) technique [39] and Class Interactions Prediction with Impact Prediction Filters (CIP-IPF) technique [38]. Whereas, Dynamic Impact Analysis (DIA) techniques considers dynamic information created by implementing the code to generate a set of potential impact classes [9]. Some of Common DIA techniques are: Path Impact technique [24] and the Influence Mechanism technique [37].

However, studies [24, 38, 39] shows that the integration of static and dynamic impact analysis as a new approach. Foundational to this, a model is developed for SDP and named as SDP Change Impact Analysis Framework (SDP-CIAF) [40]. Whereas, this framework has integrated SIA and DIA techniques and they are considered partially developed classes. Furthermore, it utilizes software artifacts such as requirement, design and class for impact analysis. In addition, the SDP-CIAF model has two important processes which are: Developing Class Interactions Prediction (CIP), and Performing Impact Analysis. The first process focuses to develop a CIP model by using requirement and design artifacts; while second process identifies the possible affected classes of the developed CIP model by using refinement techniques [41].

3. CONCEPTUAL FRAMEWORK

This research presents a new algorithmic-based Software Change Effort Estimation Model for SRCs during SDP. Figure 1 shows the conceptual model of this research.

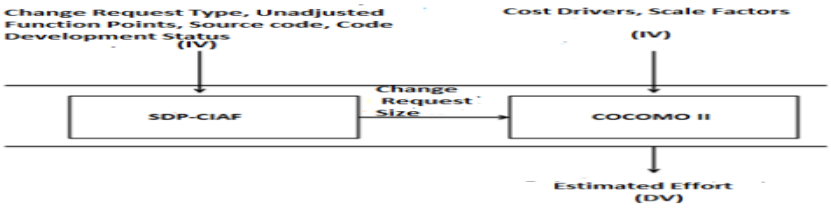

Figure 1: Conceptual model of this research

The CIA technique proposed in conceptual framework is the SDP-Change Impact Analysis Framework (SDP-CIAF) by [40] and the SCEE model proposed in the conceptual framework is the Constructive Cost Model II (COCOMO II) by [42]. The new SCEEM has combined these two techniques to increase the accuracy of change effort prediction for SRCs during SDP. The input variables of the SCEEM from SDP-CIAF are: change request type, unadjusted function points, source code, and code development status. These inputs are the independent variables (IVs) of the model. Whereas, the input variables from COCOMO II are: five scale factors and seven cost drivers [42]. Additionally, the change request size is a mediating factor and one of the inputs into the customized COCOMO II. The mediating factor positions between the independent variable (IV) and the dependent variable (DV) and it mediate the effects of the IV on the DV [43]. Finally, the estimated effort produced by the SCEEM as the dependent variable.

4. RESEARCH DESIGN AND METHODOLOGY

Research design plans the method and steps in conducting the research of collecting, analyzing and interpreting the data using quantitative, qualitative or mixed method approach. In designing the research, this research has focused on conducting research methodology specifically in software engineering field. Recently, there are a lot self-reflection in software engineering research, which have involved on what establishes a scientific discipline and discussion of empirical software engineering research [43-45]. However this research has selected the inclusive guidelines introduced by Wohlin and Aurum [45] in designing the research plan.

Wohlin and Aurum [45] Suggested decision points for designing a research, which may be put together into a decision-making structure in designing a research as specified in Figure 2.

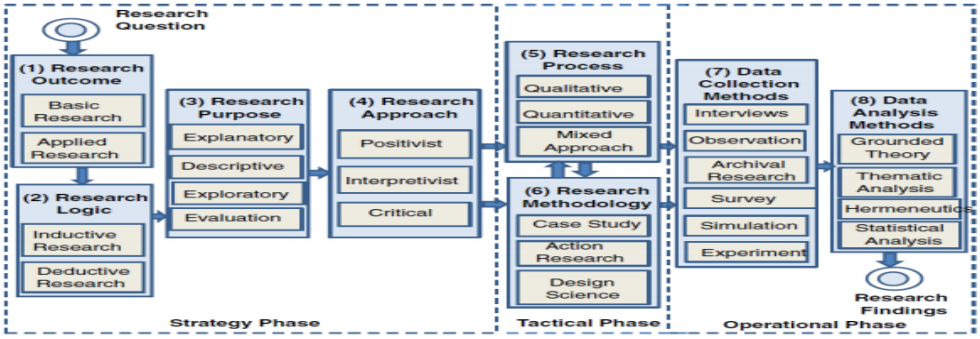

Figure 2. Research decision-making structure [45]

As shown in the Figure 2. That, there are eight (8) decision points which are dived into three (3) phases. Furthermore, every decision point can be performed by using different methods [45]. Hence, this research has used the direction of Wohlin and Aurum [45] and outlined the research decision-making structure of the research design as shown in Table 1.

Table 1. Research Design Decision

Research Phases	Decision Points	Research Method
Strategy	Research Outcome	Applied Research
	Research Logic	Deductive
	Research Purpose	Evaluation
	Research Approach	Positivist
Tactical	Research Process	Quantitative
	Research Methodology	Case Research
Operational	Data Collection Methods	Experiment
	Data Analysis Methods	Statistical Analysis

4.1. PROPOSED SOFTWARE CHANGE EFFORT ESTIMATION MODEL

Figure 3 shows the steps of the proposed Software Change Effort Estimation Model which have been executed to estimate the required amount of effort for SRCs during SDP.

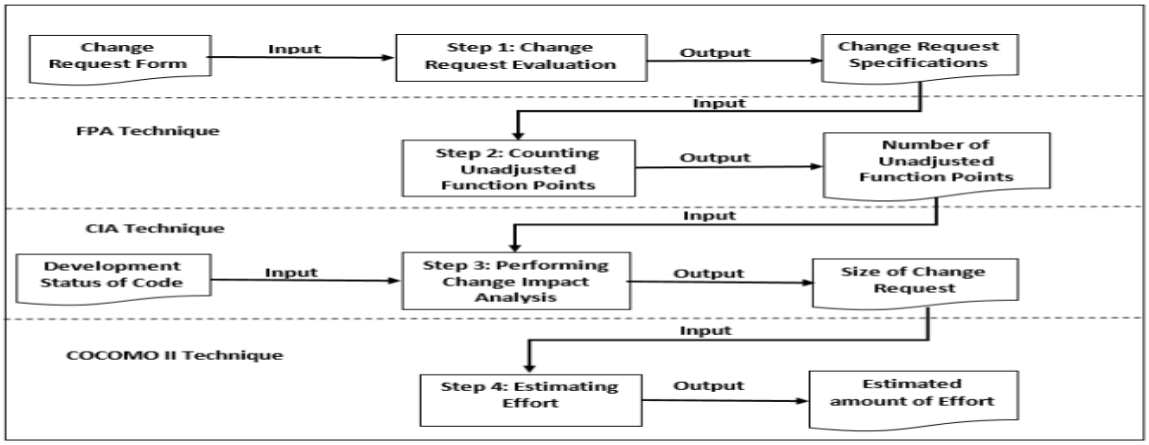

Figure 3. Software Change Effort Estimation Proposed Model

It initiates when a Software Requirement Change (SRC) occurs during Software Development Phase (SDP) and the process of change request evaluation will take place. Later, the number of Unadjusted Function Points (UFPs) will be counted using Equation 1[46, 47].

Equation 1. Unadjusted Function Points

Whereas;

- UFPs stands for Unadjusted Function Points
- ILF stands for Internal Logical Files
- EIF stands for External interface files
- EI stands for External Input
- EO stands for External Output
- EQ stands for External Inquiry

In the next step, the development status of the code (i.e. Fully Developed 100%, Major Developed 75%, Half Developed 50%, Minor Developed 25% or Not Developed 0%) will be analyzed through CIA technique and the size of software requirement change will be calculated

using Equation 2. Finally, the estimated effort for a software requirement change will be calculated in person per month unit by using COCOMO II [42] estimation model.

Equation 2. Software Requirement Change Size for Deletion or Modification Deletion

Whereas;

- CRS: stands for Change Request Size
- UFPs stands for Unadjusted Function Points can be calculated by Equation 1
- CR stands for Conversion Ratio can be calculated by [48]
- DSM: stands for Development State Multiplier see Table 2

Table 2. Code Development State Multiplier

Development State	Symbol	Multiplier	Completion %
Fully Developed	FD	1	100 %
Major Developed	MD	0.75	75%
Half Developed	HD	0.50	50%
Minor Developed	Min D	0.25	25%
Not Developed	ND	0	0%

4.2. EVALUATION FACTORS

The research has selected four evaluation factors during this research which are: (i) Subjects and Case Selections; (ii) Data Collection; (iii) Evaluation Metrics; and (iv) Evaluation Design.

4.2.1. SUBJECTS AND CASE SELECTIONS

The subjects of the experiment are two groups of Master of Software Engineering students from Universiti Teknologi Malaysia, Advanced Informatics School (UTM-AIS). Some of the team members have more than five years of field experience in software development.

The case selections are based on the following criteria:

- Small size of software projects which implemented either Traditional or Agile methodology.
- Software projects which are in the development states (requirement analysis, design, coding, testing or deployment phase).
- Software projects that implemented in any programing language.
- The development phase duration from three (3) to six (6) months.

As a case selection four software projects have been selected which are: (i) Payroll System, (ii) Vending Machine, (iii) On-Board Automobile and (iv) Software Change Effort Estimation Prototype Tool. The Table 5.1 gives a brief description of the software projects which are selected as case selection

Table 3: Case Selection Software Projects

Project ID	Project Name	Overview
P1	Payroll System (PS)	This system is developed for UTM, AIS. The system is designed in such a way to solve the different issues of existing Payroll System which was being used before in UTM AIS. In the new Payroll System employee can perform the following tasks such as: Login in the system to create employee report and maintain timecard. Whereas, Payroll Administrator can perform the following tasks such as: Login in the system to maintain employee information, create timecard and create administrative report. Furthermore, the Payroll System also deals with Bank System, System Clock, Printer and Project Database for the financial tasks.
P2	Vending Machine Control System (VMCS)	This system is developed for UTM, AIS. The purpose of the system is to vend drink cans. A Customer can perform the following tasks such as: select drink or cancel drink, pay the required amount for the drink and can collect the remaining amount if paid extra. Whereas, the maintainer can perform the following tasks such as: maintain the system, put the drinks and can collect or put the cash in the machine.
P3	On-Board Automobile (OBA)	This system is developed for UTM, AIS. The purpose of the system is to improve the safety of vehicle driving over long trips on motorways. A driver can perform the following tasks such as: Activate/ De-Activate, Auto cruise, Calibrate Speed, Monitor Maintenance, Manage Fuel and Trip. Whereas, the system informs driver

		of certain problems such as: engine broken, oil filter change, over speed indicator and refill fuel. In addition to the above, the system will provide driver with the average speed and fuel consumption information.
P4	Software Change Effort Estimation Prototype Tool (SCEEPT)	This software is developed to automate the implementation of new SCEEM. A user can perform the following tasks such as: calculate the total number of Unadjusted Function Points, to know the development state of change request, to estimate the required amount of effort for any change request. Whereas, the system will estimate and display the following results to the user such as: Total number of Unadjusted Function Points, development status of code for any change request, predicted amount effort and the value of Magnitude Relative Error between estimated effort and actual effort.

4.2.2. DATA COLLECTION

During the Data Collection phase, the following documents are collected from each case selection for the experiment.

- Change Request Form
- Software Requirement Specifications Document (If available)
- Software Design Document (If available)
- Source Code
- Progress Report (used for actual amount of effort)

The information from the above objects are collected and used as input to the Experiments to evaluate the applicability of SCEEM for SRCs during SDP.

4.2.3. EVALUATION METRIC

A set of evaluation metrics are used for the assessment of the estimation results that are produced by the SCEEM. The evaluation metrics are: (1) Magnitude of Relative Error (MRE); (2) Mean Magnitude of Relative Error (MMRE); and (3) Percentage of Prediction, PRED (.25) [49, 50].

MRE: It is a metric that is used for the estimation of absolute error of the estimated effort as compared to the actual effort. Furthermore, it is used to calculate the rate of the relative errors in both cases such as: over-estimation or under-estimation as shown in Equation 2 [49].

$$\text{MRE} = \frac{\text{abs[Actual Effort} - \text{Estimated Effort]}}{\text{Actual Effort}}$$

Equation 2. Magnitude of Relative Error

$$MMRE = \frac{100}{t}\sum_{i}^{t} MRE\ i$$

Equation 3. Mean Magnitude of Relative Error

The MRE value is calculated for each change request individually because it helps the user in knowing the deviation of the estimated effort as compared to actual effort from the new developed SCEEM. Whereas, MMRE is calculated for all the change requests once after the completion of MRE value for the selected software project. According to Jorgensen and Molokken-Ostvold [49] the MMRE value has an indirect relationship with effort estimation's accuracy that is the lower MMRE value indicates the higher accuracy rate in effort estimation.

Percentage of prediction, PRED (.25) is percentage of estimates that fall within 25 percent of the actual value. Furthermore, it states that that an estimation technique is acceptable if PRED (0.25) is at least 0.75 [50]. Percentage of prediction relation is shown in Equation 4, where K is the number of estimations where MRE value is less or equal to x and n is the total number of estimations.

$$\text{PRED(x)} = \text{k/n}$$

Equation 4. Percentage of prediction

4.2.4. EVALUATION DESIGN

The evaluation design organized in such way to answer Research Question "How much applicable is the new Software Change Effort Estimation Model for SRCs during SDP?" by performing experiment.

In this experiment the applicability of SCEEM is evaluated by estimating the change implementation effort for SRCs during SDP to answer the RQ: "How much applicable is the new software change effort estimation model for SRCs during SDP".
The hypotheses of the evaluation are:

Ho: The SCEEM is not applicable to estimate the change effort for SRCs during SDP.
Ha: The SCEEM is applicable to estimate the change effort for SRCs during SDP.

4.2.5. DATA ANALYSIS AND PROCEDURE

The table is analyzed based on the project ID, change request ID, change request type, estimated effort result, actual implementation effort and the magnitude of the relative error (MRE) between the estimation effort and the actual effort for each change request. The description of each item to be analyzed are described in Table 4.

Table 4. Data Analysis and Procedure

Project Items	Description
Project ID	A unique number for each software project.
Change Request ID	A unique number for each SRCs.
Change Request Type	Addition, Deletion and Modification
Estimated Effort	Effort estimated by SCEEM "Person per Month".
Actual Effort	Actual amount of effort that has been taken for the implementation of a software change request.
Magnitude Relative Error (MRE)	The absolute value of the relative error between the estimated effort result and the actual implementation effort in unit of percentage (%).

Accordingly, the data analysis procedure is conducted and after that the Mean Magnitude of Relative Error (MMRE) and Percentage of Prediction, PRED are calculated. Once the MMRE and PRED of the software projects are obtained, the overall MMRE value is evaluated to measure the applicability of the SCEEM for SRCs during SDP. Similarly, the experimental results are further analysed to examine the relationship between the MRE value and the change request type categorization.

5. RESULTS AND DISCUSSION

Table 5 shows the results of the software projects which are selected as case selections, which are: the estimated effort which is produced by SCEEM, actual implementation effort which is recorded during the development of software requirement change(s) and MRE value (percentage of discrepancy between estimated effort and actual implementation effort) sorted by the Project ID and Change Request ID. A total number of 81 SRCs that have been introduced from the case selection software projects during development phase; out of these 18 SRCs are introduced from Project 1 (P-1), 20 changes are introduced from Project 2 (P-2), 22 from Project 3 (P-3) and 21 are introduced from Project 4 (P-4).

Table 5. Case Selection Software Projects Experiment Results by SCEEM

Project ID	Change Request ID	Change Request Type	SCEEM Estimated Effort Man/Month	Actual Effort Man/Month	MRE Value
P-1	CR-1	Addition	0.160354121	0.18	0.109144
	CR-2	Addition	0.160354121	0.185	0.133221
	CR-3	Addition	0.160354121	0.13	0.233493

	CR-4	Modification Addition	0.160354121	0.188	0.147053
	CR-5	Modification Deletion	0.031671595	0.0329	0.037338
	CR-6	Addition	0.224521764	0.302	0.25655
	CR-7	Deletion	0	0	0
	CR-8	Addition	0.224521764	0.199	0.12825
	CR-9	Modification Addition	0.160354121	0.18	0.109144
	CR-10	Addition	0.160354121	0.23	0.302808
	CR-11	Modification Deletion	0.114525396	0.121	0.053509
	CR-12	Deletion	0.071264793	0.068	0.048012
	CR-13	Modification Addition	0.160354121	0.179	0.104167
	CR-14	Modification Addition	0.224521764	0.2	0.122609
	CR-15	Addition	0.224521764	0.35	0.358509
	CR-16	Modification Addition	0.224521764	0.188	0.194265
	CR-17	Addition	0.160354121	0.134	0.196673
	CR-18	Modification Addition	0.224521764	0.295	0.238909
P-2	CR-19	Addition	0.156885278	0.19	0.174288
	CR-20	Modification Addition	0.156885278	0.22	0.286885
	CR-21	Modification Deletion	0	0	0
	CR-22	Addition	0.156885278	0.18	0.128415

	CR-23	Modification Addition	0.219664823	0.19	0.156131
	CR-24	Deletion	0.097623735	0.09	0.084708
	CR-25	Modification Addition	0.156885278	0.13	0.20681
	CR-26	Modification Addition	0.219664823	0.18	0.22036
	CR-27	Addition	0.219664823	0.199	0.103843
	CR-28	Addition	0.156885278	0.13	0.20681
	CR-29	Addition	0.219664823	0.188	0.16843
	CR-30	Addition	0.156885278	0.14	0.120609
	CR-31	Addition	0.219664823	0.188	0.16843
	CR-32	Addition	0.156885278	0.13	0.20681
	CR-33	Addition	0.156885278	0.22	0.286885
	CR-34	Modification Addition	0.156885278	0.142	0.104826
	CR-35	Addition	0.156885278	0.18	0.128415
	CR-36	Addition	0.219664823	0.188	0.16843
	CR-37	Deletion	0.030986463	0.0322	0.037687
	CR-38	Addition	0.219664823	0.288	0.237275
P-3	CR-39	Addition	0.156885278	0.188	0.165504
	CR-40	Addition	0.318201597	0.18	0.767787
	CR-41	Modification Addition	0.318201597	0.355	0.103657
	CR-42	Addition	0.318201597	0.2	0.591008
	CR-43	Modification Deletion	0.198004737	0.189	0.047644
	CR-44	Addition	0.44553382	0.27	0.650125

	CR-45	Modification Addition	0.318201597	0.28	0.136434
	CR-46	Addition	0.318201597	0.4112	0.226163
	CR-47	Modification Addition	0.318201597	0.28	0.136434
	CR-48	Modification Addition	0.44553382	0.55	0.189939
	CR-49	Modification Addition	0.44553382	0.6	0.257444
	CR-50	Addition	0.318201597	0.199	0.599003
	CR-51	Modification Addition	0.318201597	0.399	0.202502
	CR-52	Addition	0.318201597	0.365	0.128215
	CR-53	Addition	0.44553382	0.395	0.127934
	CR-54	Deletion	0.44553382	0.413	0.078774
	CR-55	Addition	0.44553382	0.55	0.189939
	CR-56	Addition	0.318201597	0.388	0.179893
	CR-57	Addition	0.318201597	0.2	0.591008
	CR-58	Addition	0.44553382	0.55	0.189939
	CR-59	Modification Addition	0.44553382	0.435	0.024216
	CR-60	Addition	0.318201597	0.412	0.227666
P-4	CR-61	Addition	0.190858823	0.21	0.091148
	CR-62	Modification Addition	0.136311946	0.155	0.120568
	CR-63	Modification Addition	0.136311946	0.155	0.120568
	CR-64	Deletion	0.097354403	0.089	0.09387

	CR-65	Modification Addition	0.136311946	0.177	0.229876
	CR-66	Addition	0.190858823	0.3	0.363804
	CR-67	Modification Addition	0.190858823	0.218	0.124501
	CR-68	Modification Addition	0.136311946	0.154	0.114857
	CR-69	Addition	0.190858823	0.245	0.220984
	CR-70	Modification Addition	0.136311946	0.155	0.120568
	CR-71	Addition	0.367338762	0.345	0.06475
	CR-72	Modification Addition	0.136311946	0.158	0.137266
	CR-73	Modification Deletion	0	0	0
	CR-74	Modification Addition	0.190858823	0.222	0.140276
	CR-75	Addition	0.190858823	0.231	0.173771
	CR-76	Modification Addition	0.136311946	0.18	0.242711
	CR-77	Addition	0.136311946	0.17	0.198165
	CR-78	Addition	0.136311946	0.2	0.31844
	CR-79	Deletion	0.190858823	0.208	0.08241
	CR-80	Addition	0.24779752	0.312	0.205777
	CR-81	Addition	0.24779752	0.325	0.237546

The statistical analysis is conducted whereas, the histogram of the MRE values is generated to get the basic ideas of the MRE values distribution produced by the SCEEM. Figure 4 shows the histogram of the MRE Values produced by SCEEM.

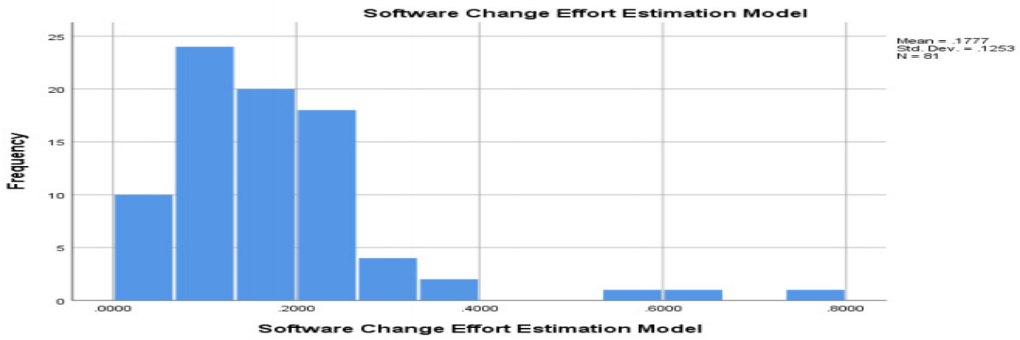

Figure 4. Histogram of the MRE Values produced by SCEEM

The histogram shows that the distribution of the MRE value produced by SCEEM are satisfactory. Whereas, the majority of the MRE values as shown in the histogram of SCEEM are below 0.25 and a few MRE values are above 0.25. This is related to the negative relationship between the MRE value and the effort estimation accuracy; the low MRE value indicates that the accuracy of the effort estimation is higher. Also, note that there is N = 81 number of MRE values, which specifies that there are no missing values in SCEEM.

Furthermore, the boxplot graph is generated to present the minimum, the lower quartile, the median, the upper quartile and the maximum quartile of the MRE value produced by SCEEM. The box of the plot is a rectangle which enclosed the middle half of the sample, with an end at each quartile. A line is drawn across the box at the sample median (Chua, 2006). The boxplot graph shows that the median value of MRE produced by the SCEEM is lower than .25. Figure 5 shows the Boxplot Graph of SCEEM.

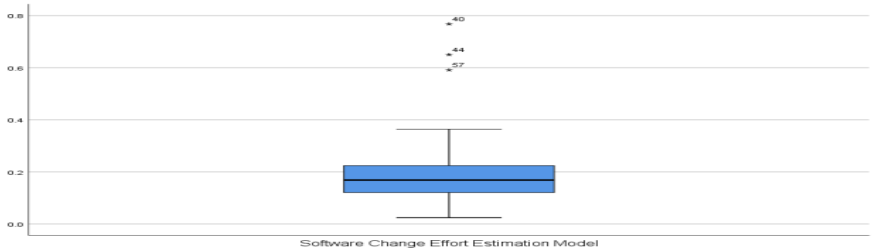

Figure 5. Boxplot Graph of SCEEM

The boxplot graph shows that the median value of MRE produced by the SCEEM is lower than .25. From the analysis, it is experimentally proven that the SCEEM gives higher accuracy in estimating the change implementation effort for requirement changes during SDP.

Moreover, the Mean Magnitude Relative Error (MMRE) of each case selection software project and overall case selection software projects are calculated. The evaluation focused on comparing results between the estimated effort with the actual effort. The MMRE and Percentage of Prediction, PRED (.25) are used as the comparison metric. According to [51], an acceptable MMRE value (or error rate) for software effort estimation is 25% or (.25). Table 6 shows the results of MMRE for each case selection software project and overall case selection software projects.

Table 6. MMRE Results of all Case Selection Projects

Project ID	MMRE	Overall MMRE	PRED (.25)
P-1	0.154091889	0.175038753	0.88
P-2	0.155724842		
P-3	0.22834514		
P-4	0.161993143		

The results of SCEEM are analyzed through statistical analysis by generating histogram, boxplot graph and finally overall MMRE value of all case selection software projects are calculated. To summarize, the experiment results which has been conducted to answer Research Question "How much applicable is the new change effort estimation model for SRCs during SDP?"

The result shows that the overall MMRE value produced by the SCEEM is under .25. Thus, it is concluded that the SCEEM is applicable in estimating the effort for requirement changes during SDP. According to [51] an acceptable MMRE value for software effort estimation is 0.25 or 25 %. Hence, the analysis of this experiment rejects the Ho and accepts the Ha of the hypothesis.

6. CONCULSION AND FUTURE WORK

This research has proposed a new Software Change Effort Estimation Model which is the combination of Effort Estimation and Change Impact Analysis. Whereas, for effort estimation this research selected COCOMO II model and for Change Impact Analysis selected SDP-CIAF. Using CIA technique in effort estimation model the proposed model generated very good results. In Table 5 it is shown that for CR-7, CR-21 and CR-73 the MRE value is zero which means that the effort estimation accuracy is 100%. This is possible only when the development status (i.e. Fully Developed, Major Developed, Half Developed, Minor Developed and Not Developed) of a software requirement change can be traced accurately. Whereas, Table 6 shows that the MMRE value of each individual project and the overall project is less than 25% which means that the new proposed model is applicable for effort estimation of SRCs during SDP [51].

The results of this research are the part and parcel of our ongoing research to overcome the challenges of accurate effort estimation for SRCs during SDP. For future work, this research is aiming to conduct an empirical study to check the effort estimation accuracy by comparing the estimation results produced from existing effort estimation model such as: FPA and COCOMO II with the new Software Change Effort Estimation Model.

ACKNOWLEDGEMENTS

This research project is under research university grant, Vote No K130000.3038.01M16 by Universiti Teknologi Malaysia

REFRENCES

[1] H. Kerzner, Project management best practices: Achieving global excellence: John Wiley & Sons, 2018.

[2] M. Gupta and A. Kalia, "Empirical Study of Software Metrics," Research Journal of Science and Technology, vol. 9, pp. 17-24, 2017.

[3] T. O. A. Lehtinen, M. V. Mäntylä, J. Vanhanen, J. Itkonen, and C. Lassenius, "Perceived causes of software project failures – An analysis of their relationships," Information and Software Technology, vol. 56, pp. 623-643, 6// 2014.
[4] R. Kaur and J. Sengupta, "Software Process Models and Analysis on Failure of Software Development Projects," CoRR, vol. abs/1306.1068, 2013.
[5] M. Bano, S. Imtiaz, N. Ikram, M. Niazi, and M. Usman, "Causes of requirement change - A systematic literature review," in Evaluation & Assessment in Software Engineering (EASE 2012), 16th International Conference on, 2012, pp. 22-31.
[6] J. Shah and N. Kama, "Issues of Using Function Point Analysis Method for Requirement Changes During Software Development Phase.," presented at the Asia Pacific Requirements Engeneering Conference, Melaka Malaysia, 2018.
[7] J. Shah and N. Kama, "Extending Function Point Analysis Effort Estimation Method for Software Development Phase," presented at the Proceedings of the 2018 7th International Conference on Software and Computer Applications, Kuantan, Malaysia, 2018.
[8] J. Shah and N. Kama, "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, pp. 77-81.
[9] Kama and M. Halmi, "Extending Change Impact Analysis Approach for Change Effort Estimation in the Software Development Phase," in WSEAS International Conference. Proceedings. Recent Advances in Computer Engineering Series, 2013.
[10] N. K. Jalal Shah, Saiful Adli Ismail, "An Empirical Study with Function Point Analysis for Software Development Phase Method," presented at the 2018 7th International Conference on Software and Information Engineering (ICSIE 2018), Cairo, Egypt, 2018.
[11] D. Kchaou, N. Bouassida, and H. Ben-Abdallah, "Change effort estimation based on UML diagrams application in UCP and COCOMOII," in 2015 10th International Joint Conference on Software Technologies (ICSOFT), 2015, pp. 1-8.
[12] D. Kchaou, N. Bouassida, and H. Ben-Abdallah, "UML models change impact analysis using a text similarity technique," IET Software, vol. 11, pp. 27-37, 2017.
[13] A. Hira, S. Sharma, and B. Boehm, "Calibrating COCOMO® II for projects with high personnel turnover," presented at the Proceedings of the International Conference on Software and Systems Process, Austin, Texas, 2016.
[14] A. Hira and B. Boehm, "Function Point Analysis for Software Maintenance," presented at the Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Ciudad Real, Spain, 2016.
[15] L. M. Alves, A. Sousa, P. Ribeiro, and R. J. Machado, "An empirical study on the estimation of software development effort with use case points," in 2013 IEEE Frontiers in Education Conference (FIE), 2013, pp. 101-107.
[16] H. Rastogi, S. Dhankhar, and M. Kakkar, "A survey on software effort estimation techniques," in Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference -, 2014, pp. 826-830.
[17] K. Usharani, V. V. Ananth, and D. Velmurugan, "A survey on software effort estimation," in 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), 2016, pp. 505-509.
[18] R. Britto, V. Freitas, E. Mendes, and M. Usman, "Effort Estimation in Global Software Development: A Systematic Literature Review," in 2014 IEEE 9th International Conference on Global Software Engineering, 2014, pp. 135-144.
[19] A. Idri, M. Hosni, and A. Abran, "Systematic literature review of ensemble effort estimation," Journal of Systems and Software, vol. 118, pp. 151-175, 8// 2016.
[20] M. d. F. Junior, M. Fantinato, and V. Sun, "Improvements to the Function Point Analysis Method: A Systematic Literature Review," IEEE Transactions on Engineering Management, vol. 62, pp. 495-506, 2015.

[21] B.Chinthanet, P.Phannachitta, Y. Kamei, P. Leelaprute, A. Rungsawang, N. Ubayashi, et al., "A review and comparison of methods for determining the best analogies in analogy-based software effort estimation," presented at the Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, 2016.

[22] S.Basri, N.Kama, F. Haneem, and S. A. Ismail, "Predicting effort for requirement changes during software development," presented at the Proceedings of the Seventh Symposium on Information and Communication Technology, Ho Chi Minh City, Viet Nam, 2016.

[23] M.Shahid and S.Ibrahim, "Change impact analysis with a software traceability approach to support software maintenance," in 2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 2016, pp. 391-396.

[24] S.Basri, N.Kama, and R. Ibrahim, "A Novel Effort Estimation Approach for Requirement Changes during Software Development Phase," International Journal of Software Engineering and Its Applications, vol. 9, pp. 237-252, 2015.

[25] O.Fedotova, L.Teixeira, and H. Alvelos, "Software Effort Estimation with Multiple Linear Regression: Review and Practical Application," J. Inf. Sci. Eng., vol. 29, pp. 925-945, 2013.

[26] A.Idri, F.a. Amazal, and A. Abran, "Analogy-based software development effort estimation: A systematic mapping and review," Information and Software Technology, vol. 58, pp. 206-230, 2// 2015.

[27] M.Kaur and S.K. Sehra, "Particle swarm optimization based effort estimation using Function Point analysis," in Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on, 2014, pp. 140-145.

[28] N.K. b. Jalal Shah*a, Amelia Zahari, "AN EMPIRICAL STUDY WITH FUNCTION POINT ANALYSIS FOR REQUIREMENT CHANGES DURING SOFTWARE DEVELOPMENT PHASE," in ASIA International Multidisciplinary Conference 2017, Johor Bharu, 2017.

[29] B.Sufyan, K. Nazri, H. Faizura, and A. I. Saiful, "Predicting effort for requirement changes during software development," presented at the Proceedings of the Seventh Symposium on Information and Communication Technology, Ho Chi Minh City, Viet Nam, 2016.

[30] V.K. Bardsiri, D.N.A. Jawawi, A. K. Bardsiri, and E. Khatibi, "LMES: A localized multi-estimator model to estimate software development effort," Engineering Applications of Artificial Intelligence, 2013.

[31] F.Ferrucci, C. Gravino, and L. Lavazza, "Simple function points for effort estimation: a further assessment," presented at the Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, 2016.

[32] A.J. Albrecht, "AD/M productivity measurement and estimate validation," IBM Corporate Information Systems, IBM Corp., Purchase, NY, 1984.

[33] P.Vickers and C. Street, "An Introduction to Function Point Analysis," School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, UK, 2001.

[34] S.Sabrjoo, M Khalili, and M. Nazari, "Comparison of the accuracy of effort estimation methods," in 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2015, pp. 724-728.

[35] V.Anandhi and R. M. Chezian, "Regression techniques in software effort estimation using cocomo dataset," in Intelligent Computing Applications (ICICA), 2014 International Conference on, 2014, pp. 353-357.

[36] B.W. Boehm, Software Cost Estimation with Cocomo II: Prentice Hall, 2000.

[37] S.Basri, N. Kama, and R. Ibrahim, "COCHCOMO: An extension of COCOMO II for Estimating Effort for Requirement Changes during Software Development Phase," 2016.

[38] Asl and Kama, "A Change Impact Size Estimation Approach during the Software Development," in 2013 22nd Australian Software Engineering Conference, 2013, pp. 68-77.

[39] B.Sufyan, K. Nazri, A. Saiful, and H. Faizura, "Using static and dynamic impact analysis for effort estimation," IET Software, vol. 10, pp. 89-95, 2016.

[40] N.Kama and F.Azli, "A Change Impact Analysis Approach for the Software Development Phase," presented at the Proceedings of the 2012 19th Asia-Pacific Software Engineering Conference - Volume 01, 2012.
[41] N.Kama and F.Azli, "A Change Impact Analysis Approach for the Software Development Phase," in 2012 19th Asia-Pacific Software Engineering Conference, 2012, pp. 583-592.
[42] B.W. Boehm, R. Madachy, and B. Steece, Software cost estimation with Cocomo II with Cdrom: Prentice Hall PTR, 2000.
[43] J.W. Creswell, Research design: Qualitative, quantitative, and mixed methods approaches: Sage publications, 2013.
[44] R.De Lemos, H. Giese, H. A. Müller, M. Shaw, J. Andersson, M. Litoiu, et al., "Software engineering for self-adaptive systems: A second research roadmap," in Software Engineering for Self-Adaptive Systems II, ed: Springer, 2013, pp. 1-32.
[45] C.Wohlin and A. Aurum, "Towards a decision-making structure for selecting a research design in empirical software engineering," Empirical Software Engineering, vol. 20, pp. 1427-1455, 2015.
[46] D.Garmus and D. Herron, "Function Point Analysis: Measurement Practices for Successful Software Projects pdf," 2001.
[47] J.J.Cuadrado-Gallego, P. Rodriguez-Soria, A. Gonzalez, D. Castelo, and S. Hakimuddin, "Early Functional Size Estimation with IFPUG Unit Modified," in Computer and Information Science (ICIS), 2010 IEEE/ACIS 9th International Conference on, 2010, pp. 729-733.
[48] Q.S.Management. (2018). Function Point Languages Table. Available: http://www.qsm.com/resources/function-point-languages-table
[49] M.Jorgensen and K. Molokken-Ostvold, "Reasons for software effort estimation error: impact of respondent role, information collection approach, and data analysis method," IEEE Transactions on Software engineering, vol. 30, pp. 993-1007, 2004.
[50] V.Nguyen, B.Steece, and B. Boehm, "A constrained regression technique for COCOMO calibration," in Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, 2008, pp. 213-222.
[51] S.-J. Huang, N.-H. Chiu, and L.-W. Chen, "Integration of the grey relational analysis with genetic algorithm for software effort estimation," European Journal of Operational Research, vol. 188, pp. 898-909, 2008.

AUTHORS

Jalal Shah, he is pursing PhD in the field of Software Engineering from Universiti Teknologi Malaysia. He has more than 8 year of teaching & research experience. He is currently working in the area of Software Effort Estimation. He has also published & presented papers in refereed journals and conferences.

Nazri Kama, He is an Associate Professor at Universiti Teknologi Malaysia (UTM) specializing in software engineering. He graduated in Bachelor in Management Information System from Universiti Teknologi Malaysia. Later, he obtained a Master Degree from the same university in Real-time Software Engineering. In 2011, he received a Doctorate in Software Engineering from the University of Western Australia in Australia

Nur Azaliah Abu Bakar, PhD is a Senior Lecturer at Advanced Informatics Department, Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia. She graduated with a Bachelor (Information Technology) in Information Systems Engineering from Multimedia University (MMU) Malaysia (2000). She then obtained her Masters in Information Technology from Universiti Teknologi Mara (UiTM) in 2004. In 2017 she was awarded a Doctor of Philosophy degree in Information Technology (Enterprise Architecture) by Universiti Teknologi Malaysia.