Efficient and Noiseless Mining Software Engineering Data using Model Approach

A Dissertation proposal submitted in fulfilment of the requirements for the Degree of

MASTER OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

GURTEJ SINGH UBHI

11607398

Supervisor

JASPREET KAUR SAHIWAL

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India)

Dec 2017

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering

Program : P172::M.Tech. (Computer Science and Engineering) [Full Time]

COURSE CODE : CSE548 **REGULAR/BACKLOG :** Regular **GROUP NUMBER :** CSERGD0037

Supervisor Name : Jaspreet Kaur Sahiwal **UID :** 14752 **Designation :** Assistant Professor

Qualification : _____ **Research Experience :** _____

SR.NO.	NAME OF STUDENT	REGISTRATION NO	BATCH	SECTION	CONTACT NUMBER
1	Gurtej Singh Ubhi	11607398	2016	K1637	9988583060

SPECIALIZATION AREA : Programming-I **Supervisor Signature:** _____

PROPOSED TOPIC : Efficient and Noiseless Mining Software Engineering Data using Model Approach

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	**Parameter**	**Rating (out of 10)**
1	Project Novelty: Potential of the project to create new knowledge	7.00
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	6.75
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	7.00
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.25
5	Social Applicability: Project work intends to solve a practical problem.	6.75
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	7.00

PAC Committee Members		
PAC Member 1 Name: Kewal Krishan	UID: 11179	Recommended (Y/N): Yes
PAC Member 2 Name: Raj Karan Singh	UID: 14307	Recommended (Y/N): NA
PAC Member 3 Name: Sawal Tandon	UID: 14770	Recommended (Y/N): NA
PAC Member 4 Name: Dr. Pooja Gupta	UID: 19580	Recommended (Y/N): Yes
PAC Member 5 Name: Kamlesh Lakhwani	UID: 20980	Recommended (Y/N): NA
PAC Member 6 Name: Dr.Priyanka Chawla	UID: 22046	Recommended (Y/N): Yes
DAA Nominee Name: Kuldeep Kumar Kushwaha	UID: 17118	Recommended (Y/N): Yes

Final Topic Approved by PAC: **Efficient and Noiseless Mining Software Engineering Data using Model Approach**

Overall Remarks: Approved

PAC CHAIRPERSON Name: 11024::Amandeep Nagpal **Approval Date:** 04 Nov 2017

11/25/2017 11:16:53 AM

DECLARATION STATEMENT

I hereby declare that the research work reported in the dissertation/dissertation proposal entitled "EFFICIENT AND NOISELESS MINING OF SOTWARE ENGINEERING DATA USING MODEL APPROACH" in partial fulfillment of the requirement for the award of Degree for Master of Technology in Computer Science and Engineering at Lovely Professional University, Phagwara, Punjab is an authentic work carried out under supervision of my research supervisor Mrs. Jaspreet Kaur Sahiwal. I have not submitted this work elsewhere for any degree or diploma.

I understand that the work presented here with is in direct compliance with Lovely Professional University's Policy on plagiarism, intellectual property rights, and highest standards of moral and ethical conduct. Therefore, to the best of my knowledge, the content of this dissertation represents authentic and honest research effort conducted, in its entirety, by me. I am fully responsible for the contents of my dissertation work.

Signature of Candidate

GURTEJ SINGH UBHI

11607398

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the M.Tech Dissertation/dissertation proposal entitled "**EFFICIENT AND NOISELESS MINIG OF SOFWARE ENGINEERING DATA USING MODAL APPROACH"** submitted by **Gurtej Singh Ubhi** at **Lovely Professional University, Phagwara, India** is a bonafide record of his original work carried out under my supervision. This work has not been submitted elsewhere for any other degree.

Signature of Supervisor

(Jaspreet Kaur Sahiwal)
Date: _____

Counter Signed by:

Concerned HOD:

HoD's Signature: _____

HoD Name: _____

Date: _____

Neutral Examiners:

External Examiner

Signature: _____

Name: _____

Affiliation: _____

Date: _____

Internal Examiner

Signature: _____

Name: _____

Date: _____

ACKNOWLEDGEMENT

I take this opportunity to present my votes of thanks to all those guidepost who really acted as lightening pillars to enlighten our way throughout this project that has led to successful and satisfactory completion of this study.
I am grateful to our **Lovely Professional Universit**y for providing me with an opportunity to undertake this project in this university and providing us with all the facilities.

I am really thankful from my heart to **Mrs. Jaspreet Kaur Sahiwal,** who allowed me to do this project under his guidance. I am highly thankful to my family and friends for their active support, valuable time and advice, whole hearted guidance, sincere cooperation and pains-taking involvement during the study.

Lastly, I thankful to all those, particularly the various friends, who have been instrumental in creating proper, healthy and conductive environment and including new and fresh innovative ideas during the project, without their help, it would have been extremely difficult for me to prepare the project in a time bound framework.

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

The Software Engineering is a vast domain that comprises the multifarious aspects of mining, efforts and fault tolerance. The mining takes the various input sets and allows to efficiently mine the effort at times when it will be with the allowance of the functions points of the projects which forms the basis of any software engineering data that imbibes the total efficiency .This help in optimization of the data set which is the combination of plethora of input ranges. The efforts are to be estimated before the beginning of the development of software. This technique is known as effort estimation. Various models are proposed to measure the efforts accurately. The proposed work includes estimating the efforts of software using hybrid technique involved in previous models. COCOMO is constructive cost model and is considered as the most accurate model for effort estimation. Another method to measure the efforts is function point. It basically measures the size of the project that further helps to calculate efforts. In this work, a hybrid formula is used that depends upon the values of cost drivers. The generated result will estimate closer efforts than COCOMO. IVR dataset is being used after taking authentication from the website. This report presents effort estimation models, COCOMO, COCOMO 2, Bailey-Baisly model, Halstead ,Bee Colony Optimization Algorithm and the proposed one. In this there are 47 projects with defined values of cost drivers.

CHAPTER 1

INTRODUCTION

Software engineering is widespread domain that allows establishing the various multifarious tasks of day to day life in which the Software Mining forms the framework of the same that includes the process of gathering and designing the information of programming and extraction of some kind of learning from it. For instance, analysts removed deployment plans from a large quantity of lines of code of the Linux portion keeping in mind the last part objective to determine bugs [1].

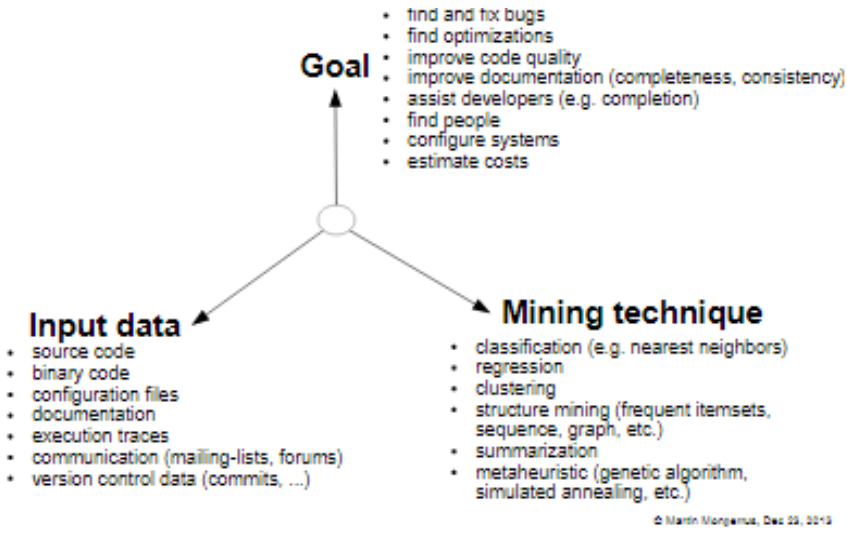

Fig 1: Fishbone Diagram for Software Artifacts.

1.1 SOFTWARE MINING

Software Mining can be defined as knowledge detection within the vicinity of software rejuvenation that employs perceptive presented artifacts. This practice is allied to a notion of reverse engineering.

Typically the knowledge obtained from presented software is accessible in the structure of models to which precise queries can be furnished as obligatory.

1.1.1 THE GOALS

The Software Engineering at the bigger level comprises of the many assignments from determination, plan, advancement, checking at dynamic time. To exemplify, a designer always amends amid task, another paradigm, navigating code, certification, composing code, investigating, along with so into view. Along the majority topical decade, that has been established that most programming designing coursework's will turnover by information mining styles, the undertakings being whether specialized [2] or more individuals arranged [3]. In addition, the faction, in spite of the fact that favoring solid commitments, likewise delivers exploratory outcomes, spellbinding marvels saw in programming building knowledge [4].

The goal further involves the estimation of the efforts on the project planning and pre planning it with the various optimization methods to reduce the real time efforts and allow it for the real time scenarios.

1.1.2 THE INPUT DATA

The input may be range from a variety of the sources. Clearly, one thinks only of code, however here are similarly numerous artifacts (details, documentation, object oriented diagrams, requirements), outline archives (graphs, recipes), and runtime reports (logs). Contingent ahead for the focused on objective, a few ancient rarities are much fitting, with mixed methodologies are plausible (using different kinds with programming building antiques in combination). Additionally, nearby be typically a decent measure of pre-handling that is particular to the antiques: characteristic dialect preparing for composed reports [5], inert reversal for code. The input may be a combination of various input points like the use case diagrams along with the variety of code

1.1.3 THE TECHNIQUES

Now a days there is multifarious methods that is being employed in the mining of same. This could be the combination of the classification, from administered to unsupervised techniques, arithmetical, group or using network, plentiful strategies have been utilized. At hand are activities, to exemplify, amalgamated data sets [6] in addition to challenges [7] to empower logical correlations of suitability and the process of the execution.

1.2 THE EVALUATION PROCESS

It is found that a lot of the software institution has gathered the huge volumes of the data with the hope of the better understanding and evaluation of their processes as well as their products. Although useful in extraction of larger set of data but at the same time plethora of valid and useful data leftovers hidden in the software engineering data warehouse and data engines along with the software repositories. To answer all those questions, the software mining has emerged as a tool for the better exploration of all such software engineering data. Software mining is firmly identified with information mining as obtainable programming antiquities enclose gigantic business esteem, key for the development of Software frameworks. Learning disclosure from software frameworks tends to structure, conduct as well as also the information handled by the product framework. Rather than mining singular informational indexes, software mining centers on metadata, for example, database patterns.

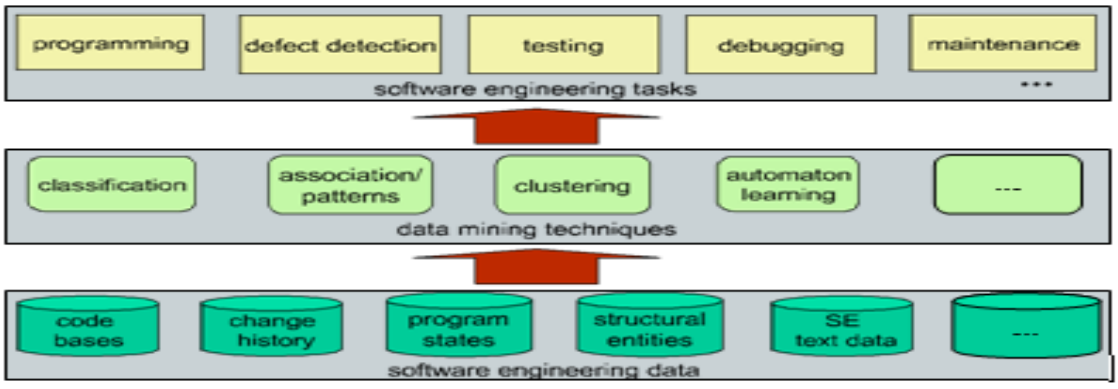

Fig 2: General idea of mining SE data

1.3 LEVELS OF SOFTWWARE MINING

It is associated to perception of reverse engineering. At the same time it looks for structure, performance as well as the information processed.

This might occur at a range of stages:

- Program level.
- Blueprint prototype level.
- Call graph level which comprises the personage procedures along with associations.
- Architectural level which is a combination of sub-systems with their interfaces.
- Data level.
- Application level.
- Business level.

1.4 SOFTWARE ENGINEERING AND KNOWLEDGE DISCOVERY

The principle aim of information mining is expectation as well as description. Prediction goes for evaluating before foreseeing the route along objective factors in view of exploration of dissimilar reasons.

1.5 SOFTWARE PROJECT ESTIMATION

In the area of software development, software project assessment is the most demanding assignment. If there is no proper and reliable estimation provided in the software development, there will be no proper arrangement as well as control of the project. Even when all the important factors are taken into consideration during the software development process still projects are not accurately estimated. It doesn't utilize estimates for improving the development of software. When a project is underestimated the effects such as under-staffing, under-scoping the quality assurance effort with missing the deadlines resulting in loss of credibility are seen [8]. Software project estimation is necessary to handle underestimates and overestimates in terms of cost, effort etc. [9]. If a project is given more number of resources than it actually requires the

resources will be utilized by it but the cost of the product increases, due to this reason deployment of estimation techniques is essential.

Small projects are not difficult to estimate and accuracy can be improved by traditional approach of Expert judgment. As the measure of project size increases i.e. for embedded and large-scale projects, precision and accuracy become important concern. Estimating the effort with a huge value of reliability is an issue which has not been unraveled yet.

1.5.1 STEPS FOR EVALUATION

Software Project Estimation is essential as already stated from the literature. For the purpose of project estimation, these four steps are considered [10]:

1) Development product's size estimation: There are Lines of Code (LOC) as well as Function Points (FP) which help in this estimation. However, various other methods are also used for measuring the estimation such as Use case points (UCP), Story points etc. There are certain merits as well as demerits of this estimation.

2) Effort estimation in terms of person-month or person-hours.

3) Scheduling estimation for months of a calendar.

4) Outlay estimation in dollars or any other local currency

1.5.2 ESTIMATING SIZE

Te first step towards achieving an effective project estimate provides a precise size approximation. Along with the formal descriptions of the requirements for project, scope of size estimation might start. Various examples of the data present for cost estimations are requirement specifications of customer, proposal requests and the specification of a system or software requirement. There can be additional details provided with the help design documents for the

chances that a re-estimation can be performed by the project at its later phases of the lifecycle [12].

For the purpose of product size estimation, the two ways are described below:

i) By analogy: For the purpose of creating a new project, it is easy to provide all the required estimations if one have already dealt with the similar kind of projects earlier. It is much likely similar to the previous one which helps in providing the total cost of the project according to its size. The total size estimates to the percentage comprising diminutive quantity of preceding project. When estimating the size of new project, the sizes of all smaller pieces are to be included. With the help of analogy, an experienced estimator can create better size estimates. Only when there are accurate size values of the previous project available and the innovative project is similar to the preceding one, only then can this technique be efficient.

ii) Including product features: along with an algorithmic approach: Function Points are for instance used as an algorithmic approach to renovate tally into an approximation of dimension. There are numerous subsystems, classes/modules, methods/functions, which are included in the macro-level "product features".

1.5.3 ESTIMATING EFFORT

After receiving the size estimation of a product, it becomes very simple for estimation the effort of it. When the SDLC of a project is defined only then the alteration from software extent to total project effort estimation is possible. Further the designing, develop and test of the software are defined for project development. In addition to coding of the software there is much more to the software development project [12]. The smallest part of the effort is basically the coding part. The larger part here includes the writing and reviewing documents, implementation of the prototypes, deliverable designing, reviewing as well as testing of the code include the larger part of the project effort. There are certain guidelines provided for the purpose of identifying, estimating as well as summing up each of the activity performed for constructing the product of certain size.

For the purpose of deriving effort from size, there are two main ways:

1) The historical data of the organization itself is the most helpful thing which helps in providing estimates of which project had what size and costs with respective to each other.

There are certain factors included here which are:
(a) Documentation of actual results using previous projects.
(b) There should be minimum one project in the past which has similar size to that of the one being newly proposed. This helps in determining estimations of projects which would cost the same with similar size.
(c) A similar development lifecycle is to be developed for developing methodology which uses similar tools, side with similar skills and knowledge which will help the novel project.

2) The most accepted and appreciated approach can be selected for the cases where there are no projects for providing help in estimating the cost and size of the projects. This case can occur if you have not gathered any of your previous projects or when the project you are developing is very new and not similar to any of the earlier projects. They are used to convert size estimate into effort estimate. They are valuable but are not that accurate as that of the own historical data of an organization and the accuracy varies as per different scenarios and application areas.

1.5.4 ESTIMATING SCHEDULE

The important pace is the determination of project agenda from the effort estimate. The numeral of people working on a project, the type of work they will do, the starting time and ending time of the project are the factors that are to be involved here. The data gathered from this step is to be laid on to the calendar schedule. This also helps in determining the way in which the work can be busted down into a proper schedule [13].

1.5.5 ESTIMATING COST

Total cost of the project can be estimated through various components such as effort, hardware plus software or rentals, travel for meeting or testing related issues, instructional classes, office area, etc. All of the enlisted factors affect the estimation of effort in one way or the other.

The total estimate of the project is also dependent in a way on the amount of cost an organization allocates to it. In some of the fields the cost allocation is not done at all and its adjustment is done by increasing the labor costs per hour. The overall costs for the software development are estimated by the software development project manager accordingly.

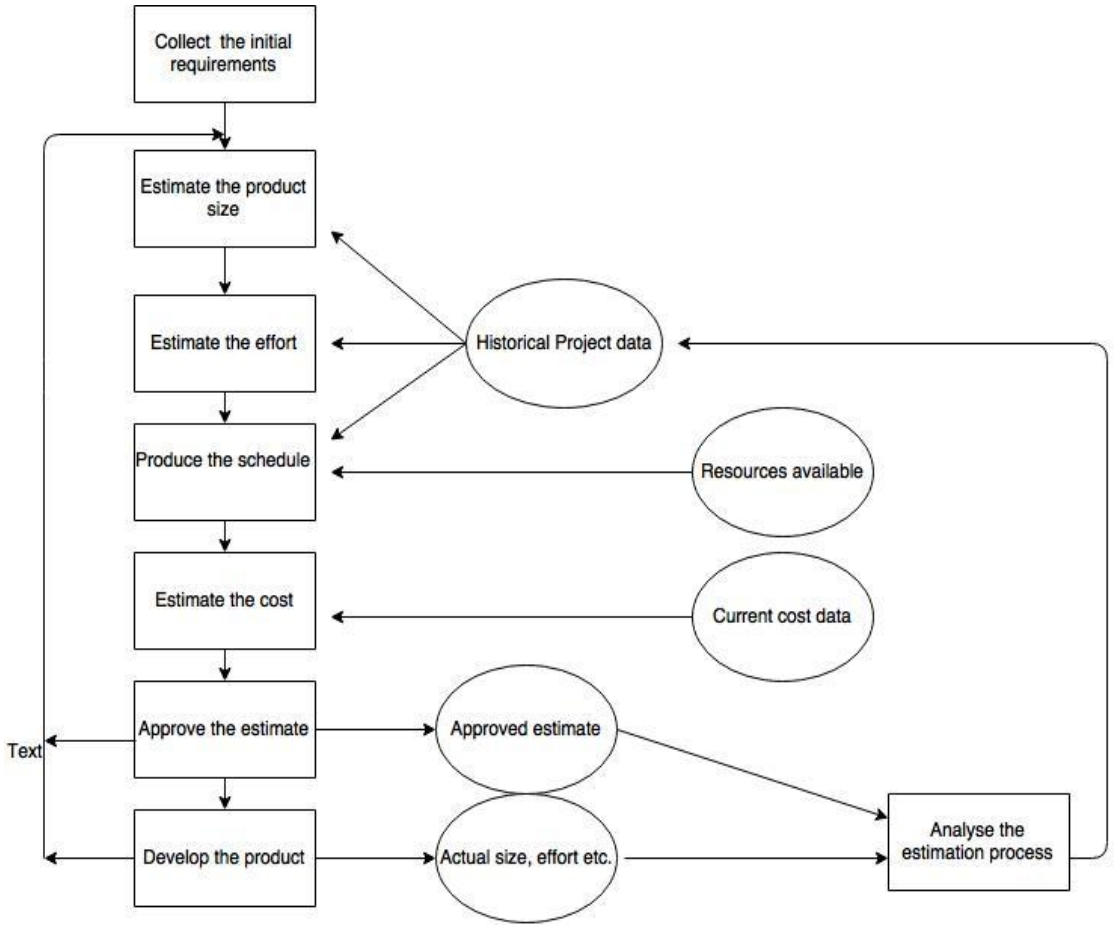

Figure 3: Software Project Estimation process

1.6 MODELS FOR EFFORT ESTIMATION

Software cost estimation model is a backhanded measure, which is utilized by software personnel to envisage the cost of a project. The development of software product shifts depending upon the earth in which it is being developed. For projects with familiar environment it is anything but difficult to foresee the cost of the project [14].

For the organization to develop a cost estimation model the following things are required.

- List important or critical cost drivers.
- Prepare a scaling model for every cost driver.
- Find projects with similar environments.
- Compare the project with previous familiar projects.
- Evaluate the project whether it is feasible inside the budget constraints.
- Incorporate the critical features in an iterative manner.

Cost drivers are those critical features which affect the project. The cost drivers may vary the cost of building a project. The most important cost driver is size of the project. Magnitude of the project is calculated in Kilo lines of code (KLOC). Function points are empirical measurement to measure size of the project.

1.6.1 CONSTRUCTIVE COST ESTIMATION MODEL (COCOMO)

COCOMO model had been projected by Barry Boehm in 1981 and is a widely adopted and accepted algorithmic cost and effort estimation model. It makes use of parameters and evaluation equation from historical software project experiences for estimation process. It employs metrics for measurement of effort and effort and scale parameters for calculating effort requisite for the project [15]. Although, COCOMO model is used largely but it has limitations for deployment in today's software development process.

CHAPTER 2

REVIEW OF LITERATURE

Several researches have been done in the recent years. A Literature Survey has been done from year 2009 to 2017. The Literature Survey is presented below in the form of four sections. The first section consists of the studies undertaken to study Artificial Neural Networks (ANN) based techniques for effort estimation. The second section elaborates work done based on Fuzzy Logic .The third studies the optimization algorithms and other estimation techniques.

2.1 ESTIMATION THAT IS BASED ON THE CONCEPT OF: ARTIFICIAL NEURAL NETWORKS (ANN)

Ali Bou Nassif et al. [16] in 2012 presented Log Linear Regression (LLP) and Multi-Layer Perceptron Models (MLP) for effort evaluation of software employed on Use case points. The result linked by two models shows their performance has been better for small projects than for large sized projects.

Ali Bou Nassif et al. [17] in 2012 projected a Treeboost Model decided on stochastic gradient boosting which employed Use case points (UCP). Software size in the form of UCP's, complexity and productivity were inputted to the Treeboost model and the results were used against the Multiple Linear Regression model. The Treeboost Model outcomes were promising. The performance metrics used are Mean magnitude relative error (MMRE), Prediction (PRED), Mean Square Error (MSE) and Median of Magnitude of Relative Error (MdMRE).

Shashank Mouli Satapathy et al. [18] in 2014 proposed a machine learning techniques based model of Stochastic Gradient Boosting (SGB) to improve the software effort estimates using class point loom. The effort variables are improved by using SGB model resulting in improved prediction accuracy.

Riyanarto Sarno, et al. [19] in 2015 presented a model to improve Constructive Cost Estimation Model II (COCOMO II) effort estimation model residing on Fuzzy Logic and Artificial Neural Network (ANN). Effort multipliers are represented using Fuzzy logic and using Gaussian Membership Function (GMF). The results belonging them proposed model reduced the error and provided a better performance Measures.

Shashank Mouli Satapathy et al. [20] in 2016 proposed an approach uses the use case points (UCP) and random forest classifier for effort estimation. In the proposed scheme, the UCP approach has estimates the UCP's considering the size, complexity, productivity and also the actual effort values of 149 projects. The random forest classifier has classified impact of each of factors. The random forest classifies features by building decision trees and it is implemented by considering various case studies.

Poonam Rijwani and Sonal Jain [21] in 2016, presented a Multilayer Feed Forward (MLFFN) Artificial Neural Network Technique for enhanced effort estimation. COCOMO 81 dataset was employed consisting of 63 software projects. The validation method used was Epochs Until tolerance level>.999. The results provided reduced MRE in MLFFN than in COCOMO.

2.2 FUZZY LOGIC BASED ESTIMATION

Attarzadeh et al. [22] in 2009 showed more improved Fuzzy Logic based sculpts. This proposed approach used two-sided Gaussian membership function to perform effort estimates. The outcome was of the projected model was better results of the significance of MMRE (Mean of Magnitude of Relative Error).

Anish Mittal et al. [23] in 2010, proposed an Enhanced Fuzzy system for enhancing estimations of COCOMO model by incorporating Triangular Fuzzy function. The results were evaluated for a firm dataset and were promising.

Monika and Om Prakash Sangwan [24] in 2017 presented an analysis of diverse machine learning methods. They relied on Artificial Neural Network, Fuzzy Logic, Analogy Based

Estimations, Genetic Algorithm and other techniques. The paper highlights relevance of each techniques depending upon its own nature and environment in which it employed.

Ali Bou Nassif et al. [25] in 2011, proposed a regression model employing Sugeno Fuzzy Inference System (FIS) approach. The results achieved correctness within stipulations of Mean magnitude relative error (MMRE). Pre and post estimations varied significantly.

2.3 Others Techniques and Optimization algorithms

David L. Gonzalez-Alvarez, et al. [26] in 2014 proposed that proteins are molecules to shape the collection of alive creatures. The proteins subsist in unlike shapes like amino-acids. These proteins help in completing different biological function in living beings and there are no such functions that could be performed without them. Due to this reason the psychiatry of proteins has befall a very important matter in biology. The recognition of preserved patterns in an arrangement of related protein sequences be able to help in achieving biological data in relation to these protein functions.

This work is dedicated to envisage ordinary pattern in sets of protein sequences [26]. These were used to appraise the performance of the projected technique. The projected technique helped in making predictions and also helped in improvement of worth of the solutions which were found by biological tools.

Chalotra et al. [27] in 2015 introduced that the target of momentum examination was applying Bee Colony Optimization (BCO) meta-heuristic way. BCO methodology a "bottom-up" way to deal with modeling where uncommon sorts of artificial agents are made by similarity with bees which are utilized to take care of complex combinatorial optimization issues. The proposed model validation was done utilizing Interactive Voice Response software venture dataset of an organization [25]. The BCO approach creates different partial arrangements and best arrangement is chosen on Mean Magnitude of Relative Error. Hereby, results acquired demonstrate that the proposed BCO based model can enhance the precision of cost estimation furthermore beat different models.

S. M. Sabbagh Jafari, F. Ziaaddini [28] in 2016, presented a meta-heuristic optimization algorithm for predicting effort estimation for developing project. The NASA dataset was used to evaluate the model. The Proposed model optimized the Mean Magnitude Relative Error (MMRE) to nearly 21%.

Peyman Khazaei, et al. [29] in 2016 proposed that in the day-ahead power systems scheduling, system operators formulated and solved the unit commitment (UC) problem to determine ON/OFF status and power dispatch of the producing units. Although different methods had been exhibited to solve the UC problem, it was a blended integer optimization problem which was elusive its global optimum solution. The TLBO not just gave a solution bring down operating costs, additionally had a lower computation time. In addition, adequate spinning reserve was given to alleviate the effect of rapid load/generation changes because of unexpected disturbances.

Yu-Huei Cheng [30] in 2016 proposed a feasible PCR-RFLP primer match was to be designed. Also there was a need to discover accessible restriction enzymes which could perceive the target SNP for PCR experimental purposes. It was developed to improve mutagenic primer and it used the idea of "teaching-learning" for searching more possible mutagenic primers. It gave the latest accessible restriction enzymes.

Dragicevic Srdjana, et al [31] in 2017 proposed a Bayesian network model for predicting software endeavor assessment for agile software maturity .The future model predicts effort irrespective of the agile methods available and the input data is elicited easily. The precision and prediction accuracy achieved by the proposed Bayesian network are very favorable.

Seacord et al, This approach [32] is beneficial as with eternally escalating number of the computers and their computations in on increase at an alarming rate, it is found that an estimate of 250 billion LOC be uphold in 2000 and it is still increasing [33]. This makes the use of the Ontology based program comprehension way. As with the software age, the crucial task of maintenance of software is becoming more complex as well as crucial. Software evolution often regarded as software maintenance has the preponderance of the entirety price tag that occurs

through the life extent of the software organization [33, 34]. The safeguarding challenges is a result of the different inter-relationships and the representations the exist among the all software knowledge resources [35, 36]. A vital ingredient of construction is a software ontology that collects chief concept plus dealings in the software safeguarding sphere of influence. Following picture shows the integrated approach.

João Caldeira et al ,Due to dependence of the software in our on a daily life, the development of the same has been a crucial part in our lives.But at the same time the process is not formalized which means the model to it is not available. This uses the following set of methodologies [37].

i. Process delivery in the Software development.

ii. Conformance Checking

iii. Process Enhancement

Cagatay Catal et al, In a software enlargement project, valuable data are formed along with stored in several repositories. The authors in [38] have used the sources in order to organize repositories, bug repositories, archived communications, deployment logs, and code repositories. Instead of lying for preceding experiences, managers or else developers may investigate the data that is positioned in these repositories, to steer for verdict processes within corporation.

Calero et al, In [39] the authors have used the concept of state of art as a landscape to distinguish between various types of categories and they have made it vital in the form of the SET theory and allowed the various methods of classification and that has enabled to play a important role in artifact in the software process. They further divided them in order to attain a stable repository

Menzies et al, The authors in [40] compelled the scope of conceivable practices characterized for an area. At the point when parts of a model are unverifiable, the conceivable practices might be an information cloud: i.e. a staggering scope of potential outcomes that confuse an investigator. Looked with huge information mists, it is difficult to exhibit that a specific choice prompts a specific result. Regardless of whether we can't settle on unequivocal choices from

such models, it is conceivable to discover choices that decrease the fluctuation of qualities inside an information cloud. Additionally, it is conceivable to change the scope of these future practices to such an extent that the cloud consolidates to some enhanced mode. Our approach utilizes two instruments.

The above sections presented literature review of Software effort estimation techniques that have been employed using machine learning methods and of optimization methods. The machine learning algorithms have outperformed and provide better accuracy due to their learning natures. ANN has been used dominatingly in effort estimation models but have complex algorithms and need to be optimized

CHAPTER 3

PROBLEM DEFINITION

3.1 SCOPE OF STUDY

In the past years, many techniques have been designed for the effort estimation which are model based techniques, neural networks based techniques, use case based techniques etc. The technique mentioned in [18] is a hybrid technique of Use case and neural networks for the effort estimation. The Random Forest (RF) classifier is applied which will classify the parameters of the software. The software parameters which are analyzed using the use case are given input to the classifier algorithm. The proposed hybrid technique is efficient and gave high accuracy in terms of effort estimation measures of Mean Magnitude Relative Error (MMRE). An improvement be required in hybrid technique due to two reasons. The Random Forest is the ensemble learning model which takes input from the use case and drive relation between the parameters to use them for effort analysis of software. The complexity of random forest is high which leads to increased execution time. The second reason is accuracy; the Random Forest algorithm works with the single iteration for the classification which reduces accuracy of effort estimation as single iteration may not be sufficient validation method. The Improvement in the RF and use case technique is required which has less execution time and high accuracy.

CHAPTER 4

OBJECTIVES OF THE STUDY

The central objectives of the study are as follows:

1. To study and analyze prominent effort estimation technique like Random Forest and Teaching Learning Based Optimization.

2. To augment random forest technique using Teaching Learning Based Optimization.

3. To compare pre and post treatment performance of random forest technique for estimating software effort.

CHAPTER 5

PROPOSED RESEARCH METHODOLOGY

5.1 USE CASE POINTS (UCP)

The UCP method was anticipated by Karner [41] in the year 1993. This approach is an expansion of Function Point Analysis (FPA) and Mk II FPA. If the plan concerning about predicament domain, system size with style is lucid, next a premature effort estimation focused around use cases might be prepared.

The UCP method can be implemented by the subsequent steps:

1. Classification involving actors plus use cases.
2. Computation involving weights with points.
3. Calculation of technical complexity factor (TCF) along with environmental factor (EF).
4. Ultimate UCP calculation.

5.2 TEACHING LEARNING BASED OPTIMIZATION (TLBO)

This is employed for the optimization. In this research TLBO algorithm is functional to reduce MRE value of the hybrid model (use case + random forest) by estimating predicted efforts more accurately. The TLBO algorithm comprises of methods which help each individual to take in something different and to enhance himself. The base of this algorithm has derived from a normal teacher-earner methodology of a classroom.

The TLBO algorithm holds the basics of traditional learning methods that are seen in a teacher and a learner. There are two essential methods of learning involved in it. The first is to learn through the teacher. It is also known as a teacher phase. The second is the learning that is done through interaction with different learners. This is recognized like the learner phase. It is a population based algorithm. The population comprises of the gatherings of students (learners). The diverse subjects offered to the learners are analogous with the distinctive design variables of

the optimization issue. The consequences of the learner that are obtained are similar to the fitness value of the optimization issue. The teacher is held to be the best arrangement in the whole population. The operation of the TLBO algorithm is clarified underneath with the teacher phase with learner part.

a. Teacher Phase: This imitates the knowledge of the students (i.e. learners) through the teacher. A teacher passes on information amid the learners in this phase. The teacher tries to build the mean consequence of the category.

b. Learner phase: This imitates the learning of the students (i.e. learners) via association amid themselves. The information gathered by the students can also be from the examinations or interactions with other students. A learner will learn fresh information if alternate learners have added information than him or her.

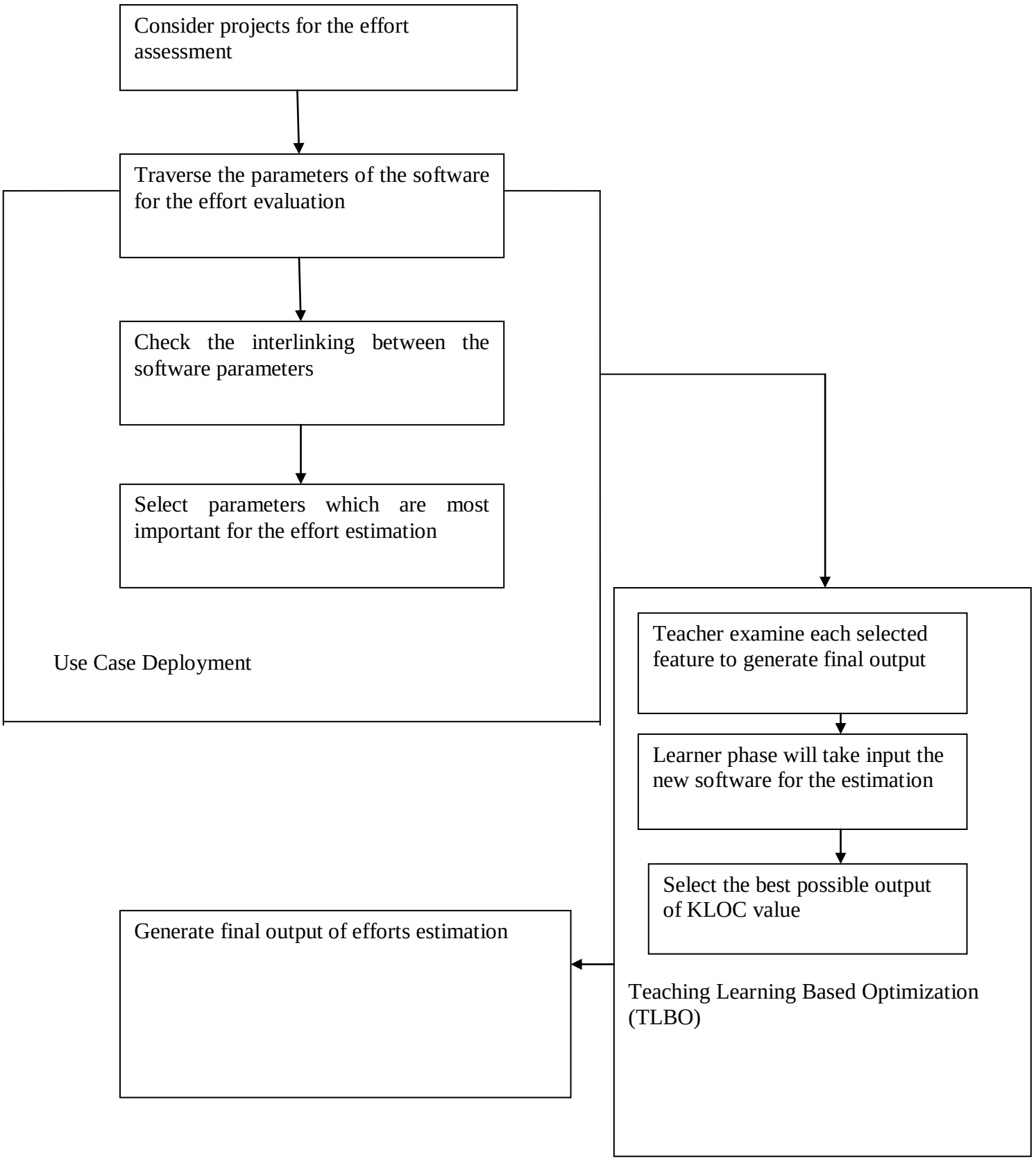

Figure 4: Proposed Flowchart

Chapter 6

EXPECTED OUTCOMES

Following are the expected outcomes:-

1. The proposed algorithm is based on to increase performance of the COCOMO model for effort estimation. This leads to increase of effort estimation in software which will in turn help to pre mine the data more efficiently.

2. The proposed improvement can also lead to reduce execution time.

.

CHAPTER 7

SUMMARY AND CONCLUSIONS

The effort estimation is the technique which is applied to estimate software efforts. The COCOMO modal works with the KLOC value. The efforts are directly proportional to KLOC; it means that when the KLOC is not analyzed properly then efforts are not estimated correctly. The TLBO algorithm will be applied with the COCOMO modal for the effort estimation. The proposed improvement can lead to increase estimation of KLOC value. The performance of the proposed technique can estimate to improve MSE value. This in turn will result to emphasize the more efficient software mined with less prone to noise using the approaches defined.

LIST OF REFERENCES

[1] Zhenmin Li, Yuanyuan Zhou. *PR-Miner: automatically extracting implicit programming rules and detecting violations in large software code.* In « Proc. of the 10th European Software Engineering Conference held jointly with 13th International Symposium on Foundations of Software Engineering », ESEC/FSE-13, ACM, 306–315, 2005.

[2] Thomas Zimmermann, A. Zeller, P. Weissgerber, S. Diehl. Mining version histories to guide software changes. In « IEEE Transactions on Software Engineering », 6, 31, June, 2005, 429–445, http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1463228

[3] D. Schuler, T. Zimmermann. Mining usage expertise from version archives. In « MSR », 2008.

[4] Daniel M. German. An empirical study of fine-grained software modifications. In « Empirical Softw. Engineering », 3, 11, September, 2006, 369–393.

[5] Stefan Henss, Martin Monperrus, Mira Mezini. Semi-Automatically Extracting FAQs to Improve Accessibility of Software Development Knowledge. In « Proceedings of the International Conference on Software Engineering », 793 - 803, 2012, http://hal.inria.fr/docs/00/68/19/06/PDF/paper.pdf

[6] Tim Menzies, Bora Caglayan, Ekrem Kocaguneli, Joe Krall, Fayola Peters, Burak Turhan. *The PROMISE Repository of empirical software engineering data*. June, 2012, http://promisedata.googlecode.com

[7] Alberto Bacchelli. *Mining Challenge 2013: Stack Overflow.* In « The 10th Working Conference on Mining Software Repositories », to appear, 2013

[8] FIONA WALKERDEN, ROSS JEFFERY," An Empirical Study of Analogy-based Software Effort Estimation", 1999, Empirical Software Engineering, 4, 135–158

[9] KJETIL MOLØKKEN-ØSTVOLD, MAGNE JØRGENSEN," Group Processes in Software Effort Estimation", 2004, Empirical Software Engineering, 9, 315–334.

[10] Leungh, Zhangf, ―_Software cost estimation ‖ in _Handbook of software engineering and knowledge engineering' (World Scientific Pub. Co, River Edge, NJ, 2001) .

[11] Li, J., Ruhe, G., Al-Emran, A., & Richter, M. M. (2007). A flexible method for software effort estimation by analogy. *Empirical Software Engineering*, *12*(1), 65-106.

[12] Yang, Y., & Laird, L. (2016, April). Teaching Software Estimation through LEGOs. In *Software Engineering Education and Training (CSEET), 2016 IEEE 29th International Conference on* (pp. 56-65). IEEE.

[13] Ren, A., & Yun, C. (2013, November). Research of Software Size Estimation Method. In *Cloud and Service Computing (CSC), 2013 International Conference on* (pp. 154-155). IEEE.

[14] Madheswaran, M., & Sivakumar, D. (2014, July). Enhancement of prediction accuracy in COCOMO model for software project using neural network. In *Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on* (pp. 1-5). IEEE.

[15] Boehm, B. W., Madachy, R., & Steece, B. (2000). *Software cost estimation with Cocomo II with Cdrom*. Prentice Hall PTR.

[16] Nassif, A. B., Ho, D., & Capretz, L. F. (2013). Towards an early software estimation using log-linear regression and a multilayer perceptron model. *Journal of Systems and Software*, *86*(1), 144-160.

[17] Nassif, A. B., Capretz, L. F., Ho, D., & Azzeh, M. (2012, December). A treeboost model for software effort estimation based on use case points. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on* (Vol. 2, pp. 314-319). IEEE.

[18] Satapathy, S. M., Acharya, B. P., & Rath, S. K. (2014). Class point approach for software effort estimation using stochastic gradient boosting technique. *ACM SIGSOFT Software Engineering Notes*, *39*(3), 1-6.

[19] Sangwan, O. P. (2017, January). Software effort estimation using machine learning techniques. In *Cloud Computing, Data Science & Engineering-Confluence, 2017 7th International Conference on* (pp. 92-98). IEEE.

[20] Satapathy, S. M., Acharya, B. P., & Rath, S. K. (2016). Early stage software effort estimation using random forest technique based on use case points. *IET Software*, *10*(1), 10-17.

[21] P. Rijwani and S. Jain. "Enhanced Software effort estimation using Multi Layer Feed Forward Artificial Neural Network Technique," Procedia Computer Science, vol 89, pp. 307-312.

[22] Attarzadeh, I., & Ow, S. H. (2009). Software development effort estimation based on a new fuzzy logic model. *International Journal of Computer Theory and Engineering*, *1*(4), 473.

[23] Mittal, A., Parkash, K., & Mittal, H. (2010). Software cost estimation using fuzzy logic. *ACM SIGSOFT Software Engineering Notes*, *35*(1), 1-7.

[24] Sangwan, O. P. (2017, January). Software effort estimation using machine learning techniques. In *Cloud Computing, Data Science & Engineering-Confluence, 2017 7th International Conference on* (pp. 92-98). IEEE.

[25] Nassif, A. B., Capretz, L. F., & Ho, D. (2011, November). Estimating software effort based on use case point model using sugeno fuzzy inference system. In *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on* (pp. 393-398). IEEE.

[26] Gonzalez-Alvarez, D. L., Vega-Rodriguez, M. A., & Rubio-Largo, A. (2015). Finding patterns in protein sequences by using a hybrid multiobjective teaching learning based optimization algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, *12*(3), 656-666.

[27] Chalotra, S., Sehra, S. K., Brar, Y. S., & Kaur, N. (2015). Tuning of COCOMO model parameters by using bee colony optimization. *Indian Journal of Science and Technology*, *8*(14).

[28] Jafari, S. S., & Ziaaddini, F. (2016, March). Optimization of software cost estimation using harmony search algorithm. In *Swarm Intelligence and Evolutionary Computation (CSIEC), 2016 1st Conference on* (pp. 131-135). IEEE.

[29] Khazaei, P., Dabbaghjamanesh, M., Kalantarzadeh, A., & Mousavi, H. (2016, July). Applying the modified TLBO algorithm to solve the unit commitment problem. In *World Automation Congress (WAC), 2016* (pp. 1-6). IEEE.

[30] Cheng, Y. H. (2016). A novel teaching-learning-based optimization for improved mutagenic primer design in mismatch PCR-RFLP SNP genotyping. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, *13*(1), 86-98.

[31] Dragicevic, S., Celar, S., & Turic, M. (2017). Bayesian network model for task effort estimation in agile software development. *Journal of Systems and Software*, *127*, 109-119

[32] R. Witte , Q. Li , Y. Zhang :" Text mining and software engineering: an integrated source

code and document analysis approach", IEEE, 2008,p.3-16.

[33]Sommerville, I.: „Software engineering" (Addison-Wesley, 2000, 6th edn.)

[34] Seacord, R., Plakosh, D., and Lewis, G.: „Modernizing legacy systems: software technologies, engineering processes, and business practices" „SEI series in SE" (Addison-Wesley, 2003)

[35] Storey, M.A., Sim, S.E., and Wong, K.: „A collaborative demonstration of reverse engineering tools", ACM SIGAPP Appl. Comput. Rev., 2002, 10, (1), p. 18–25

[36] Welty, C.:Augmenting abstract syntax trees for program understanding". Proc. Int. Conf. Automated Software Engineering, 1997, (IEEE Comp. Soc. Press), pp. 126–133

[37] João Caldeira,:" Software Development Process Mining: Discovery, Conformance Checking and Enhancement", (2016 10th International Conference on the Quality of Information and Communications Technology)

[38] Software mining and fault predictionWIREs Data Mining Knowl Discov 2012, 2: 420–426 doi: 10.1002/widm.1067

[39] Calero, C., Ruiz, F., and Piattini, M.: 'Ontologies for software engineering and software technology' (Springer-Verlag, Berlin, Heidelberg, 2006)

[40]T. Menzies, E. Chiang, M. Feather, Y. Hu and J.D. Kiper, Condensing uncertainty via incremental treatment learning. In book chapter in: Software Engineering with Computational Intelligence, T.M. Khoshgoftaar, ed., page Volume 731.Kluwer Academic Publishers, 2003.

[41] Karner, G. (1993). Resource estimation for objectory projects. *Objective Systems SF AB, 17*.