Longitudinal Collaborative Filtering With Attribute Maximization Clustering On Software Process Improvement

Dr.A.Saranya
Assistant Professor
Department of Computer Application
V.V.V College for Women Virudunagar

Abstract - Among several characteristic that impact the success of a software project, the software process model employed is a crucial one. An inappropriate process framework will not only consume too much of time but also reduces the software quality. Hence, selection of a suitable software process framework is a very critical problem for software development. Current works focus on the software process appraisals with absorptive capability and therefore compromises the performance and process improvement related to longitudinal data. In this paper, we propose a software process enhancement framework to assists project managers select the most suitable software process framework at an early stage of development process, called, Collaborative Filtered Maximization Clustering and Longitudinal Regression (CFMC-LR). The CFMC-LR framework is split into three different stages, namely, pre-processing, clustering and application of machine learning model. Pre-processing or the most relevant project attributes are first extracted by applying Contingency Matrix Collaborative Filtering technique. Next, clustering of the pre-processed project attributes into similar attributes belonging to a particular group or project is performed using Expectation Attribute Maximization Clustering algorithm. Finally, machine learning technique, Longitudinal Regression is applied to analyze the effect of Software Process enhancement. The results demonstrate that CFMC-LR framework success is appreciably impacted by Contingency Collaborative Filtering algorithm through Linear Regression model and therefore influences firm performance.

Keywords - Collaborative Filtered, Maximization Clustering, Longitudinal Regression, Machine Learning, Expectation Attribute

---

1. Introduction

The software process enhancement is popularly identified as a prime aspect that bestows to the software quality. To design software processes enhancement, reference models are used, as they provide universally set of accepted best practices for the creation of such processes Software process enhancement model is a method both by which process are said to be improved and ensure betterresult rather than a normal process. So, by software process enhancement, a better and high quality product can is said to be foundwithin budget and time.

In Process Mining Extension to SCAMPI [1], the limitations of the current Standard Capability Maturity Model Integration Appraisal Method for Process Improvement (SCAMPI) method were reduced with the application of a feasible and usable method with the aid of process mining techniques. This was said to be achieved using a four step model. In the first step, the needs, expectations and limitations were analyzed. Followed by which, the most possible assessable elements were identified. Next, the identified elements, the relationship with the process mining were derived. Finally, the best alternatives were arrived for conducting mining. Based on the four step model, the contributions of Process Mining Extension to SCAMPI were categorized into two aspects, process and content. Despite contributions made by SCAMPI in terms of process and content, judgment of CMMI practices relevant to performance and process improvement aspects remained unaddressed. Potential solution addressed in the work is the design of process improvement by performing pre-processing using Contingency Matrix Collaborative Filtering technique through the Neighbor Pearson Correlation Coefficient.

Capability Maturity Model Integration-based Software Process Improvement (CMMI-based SPI) [2] used capability theory based on dynamic manner to investigate the potentiality of an establishment to obtain knowledge pertinent to external factor and achieve SPI. Specifically, in [2], a research model based on the relationships between an establishment's Potential Absorptive Capability (PAC), Realized Absorptive Capability (RAC) were analyzed. Also, by applying PAC through cross functional interfaces, it had higher influence of firm performances. The CMMI-based SPI method when tested using partial least squares structural equation modelling technique indicated that the success of SPI was highly influenced by PAC through RAC. Despite success rate ultimately influencing the performance of firm, the relationships of the proposed method with higher amount of changes in the organization occurring over time was not addressed. Potential solution addressed in this work is the design of

machine learning technique using linear regression that addresses issues related to longitudinal data to provide more comprehensive research perspectives.

Several factors influence the understand ability of process models. A systematic review was presented in [3]. A hierarchical decision model was investigated in [4] with Business Process Simulations Software to meet the objectives of company business process.Several research works were conducted on analyzing the software product artifacts. However, to provide software process visualization, Concept Visualization was introduced in [5] to meet out the developers and managers to explore multiple aspects of software product results in a synchronized manner. To enhance software quality, a software process model based on Entity Component System was built in [6].

Software Process Requirements has long been identified as an important quality of a well-engineered system. Among researchers, Software Process Requirements is often unpopular due to the unclear benefits. Therefore, only little evidence exists regarding the expected traceabilitybenefits. In [7], Multi Level Progression Analysis was designed with the objective of identifying the defect rate while performing software process requirement. On the other hand, context aware design was presented in [8] to assist programmers in software designing using Context Aware Recommendation Algorithms. A training process was conducted in [9] with the objective of selecting appropriate software projects.

In this paper, a new Collaborative Filtered Maximization Clustering and Longitudinal Regression (CFMC-LR) framework is proposed forperforming effective software process improvement. The proposed framework introduces a new technique called Contingency Matrix Collaborative Filtering which reduces the Software development cycle time by removing the irrelevant and redundant project attributes without increasing the delay. For this purpose, new computational techniquesare introduced in this framework for identifying the correlation between the two programmers for extracting the project attributes.

Moreover, the optimal pre-processed project attributes is obtained in thiswork by applying the Contingency Collaborative Filtering algorithm to find the best project attributes using a similarity function.Moreover, project attribute clustering is used in this work for dividing the pre-processed attributes into groupsand observed prospect cluster by using Expectation Attribute Maximization Clustering algorithm. The major advantage of the proposed Expectation Attribute Maximization Clustering algorithm includes the increase in project attribute clustering accuracy, success rate and efficient selection of software process model using Linear Regression model. The rest of the paper is organized as follows:Section 2 provides a survey of software process models used by several organizations in diversified fields. Section 3 discusses the proposed algorithm and software process framework. Section 4 provides the experimental setup and discussion is provided in detail in Section 5. Section 5 gives some conclusions on this proposed work.

2. Related works

In order to increase the company competitiveness, one of the most important factors to be considered is the Customer Knowledge (CK). Therefore, research on CustomerKnowledge Management (CKM) is an ongoing topic. In [10], software quality was enhanced based on CKM in software companies. The field of software process analysis is widening with the increasing availability of software process. In [11], software process for computational neuroscience was presented by categorizing software projects and correlating them with the results. On the other hand, CMMI-based implementation was investigated in [12] to enhance the success rate.

In software process identification, continuous delivery (CD) is a software release process helps in assisting bug related to new enterprise application (EA) versions. In [13], resource profiles were used that explained the requirements of request demand for each component that in turn helped in efficiently supported capacity planning for new deployments.

Both the processes involved in the business and software face several modifications during their life cycle. This is because of the higher influential impact between the business process and software process, both have to be updated in co-evolution. In [14], a tool supported approach was presented with the objective of measuring the propagation of changed caused by business or software process. With the precision, recall and accuracy, the impact analysis was made, which then formed the basis for software process selection. Case studies for agile method adaptation were presented in [15]. A brief comparative review of several software management tools were presented in [16]. A study on the performance level, both qualitative and quantitatively managed was provided in [17].

Continuous process improvement is one of the main constituent in web-based projects. In [18], a new maturity model was designed by learning through the experience gained by other process was explained. This in turn had a positive impact where critical success factor was used to analyze software project process. Yet another software process specification for specialized geo spatial database was presented in [19]. In [20], an initialunderstanding of the project with steps considered while developing project and presenting the results of mapping between software process models was presented.

In spite of the presence of all the above said methods, the existing methods are not able to provide optimal selection of software process in the presence of several software process models. Hence, a new software process enhancement framework is designed and implemented in this work for enabling the project manager select the software process at an early stage of project development. This paper presents an effective machine learning based software process model to reduce the software process

life cycle time and improve the success rate. Moreover, this framework providesbetter performance than the existing methods in terms of the software process development life cycle time, firm cost and success rate.

3. Collaborative Filtered Maximization Clustering and Longitudinal Regression

In this paper, we propose a new framework named Collaborative Filtered Maximization Clustering and Longitudinal Regression which is used to improve the software process enhancement rate by modifying the existing absorptive capability and process miningtechniques to assist project managers in determining which software process model is more appropriate at an early stage. The proposed framework provides several advantages such as project attribute selection time, project attribute clustering accuracy and success rate at the early stage of development process. Figure 1 shows the block diagram of Collaborative Filtered Maximization Clustering and Longitudinal Regression framework.

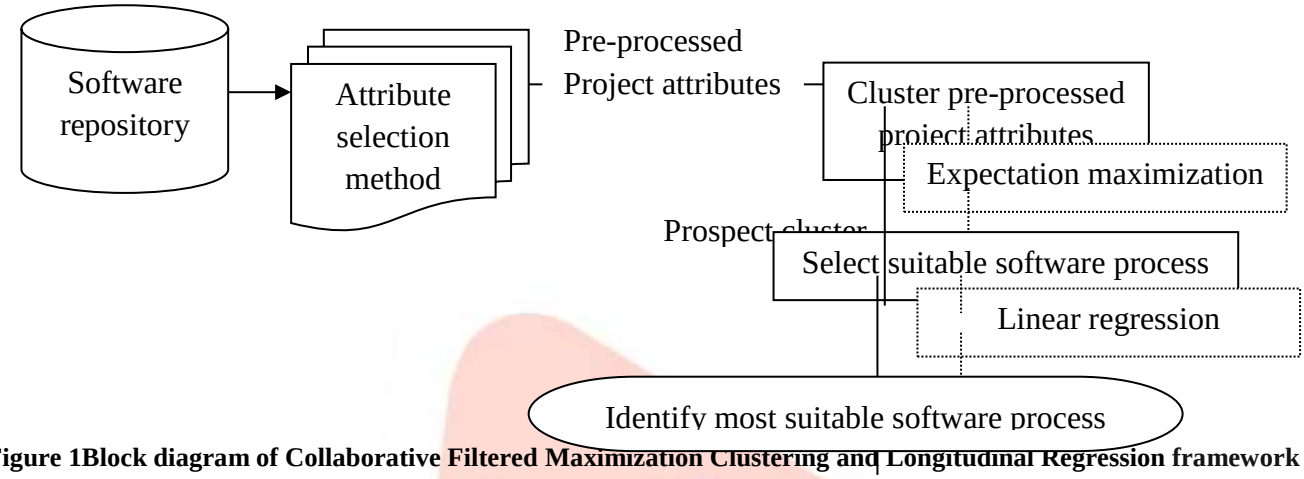

Figure 1Block diagram of Collaborative Filtered Maximization Clustering and Longitudinal Regression framework

As shown in the figure, machine learning technique using Linear Regression is used for identifying the most suitable software process model from the available software process models and to improve the success rate with less time. It is also used to find all probable project attributes in and between the clusters for effective clustering. This framework works by identifying the relevant project attributes, forming cluster, time taken for project attribute selection and success rate. This proposed CMFC-LR framework has been implementedusing JAVA platform and proved that the proposed framework provides better performance when it is comparedwith the other existing methods.

The CMFC-LR framework initially selects the project attributes that have the most effect on designing by applying selection methods. With the project attributes selected for further processing, the CMFC-LR framework then uses anExpectation Attribute Maximization Clustering algorithm via testing them with sample datasets. Followed by this, the selected Expectation Attribute Maximization Clusteringalgorithms are then used to construct machine learning technique, which finally is employed to select the most appropriate process model.

3.1 Contingency Matrix Collaborative Filtering technique

The first step of the proposed framework involves pre-processing. In this pre-processing stage, the most relevant attributes (i.e. project attributes) are selected. Due to the presence or irrelevant or redundant project attributes, identifying the most appropriate software process that helps in help project managers for a new project at an early stage becomes a complicated task. Hence, the first step towards the software process enhancement remains in identifying the most relevant attributes through pre-processing.

Let us consider a software process problem, then the corresponding data is represented in the form as '$Proj_{ID}, Attr_1, Attr_2, \ldots., Attr_n$'. Here '$Proj_{ID}$' represents the software project and '$Attr_1$' equates to the attribute that identifies one characteristic of the software project. In the proposed work, the pre-processing is said to be performed by applying Contingency Matrix Collaborative Filtering technique. Figure 2 shows the contingency matrix to perform collaborative filtering.

	Att_1	Att_2	Att_j	Att_n
$Prog_1$				
$Prog_2$				
$Prog_3$				
……				
$Prog_i$				
$Prog_n$				

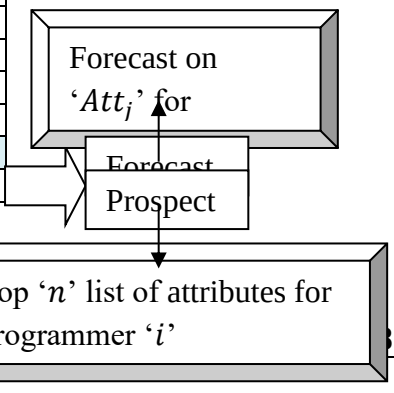

Figure 2 Block diagram of Contingency Matrix Collaborative Filtering technique

Contingency Matrix Collaborative Filtering techniqueas given in the above figure involves a domain-independent forecasting technique. The Contingency Matrix Collaborative Filtering technique works by constructing a matrix, where the rows represents the programmer involved '$Prog$', and the column represents the project attributes 'Att', therefore forming a programmer-attribute matrix of preferences for attributes by programmers. It thenmatches programmers with relevant preferences by measuring the similarities between their profiles to make prospects, based on the neighborhood.

The similarities are obtained using the resultant prospects to those attributes that he has not ratedbefore but that were already positively rated by other programmers in hisneighborhood. Pearson Correlation Coefficient is used in the proposed framework of measure the similarities based on the neighborhood and is mathematically formulated as given below.

$$S(Prog_P, Prog_Q) = \frac{\sum_{i=1}^{n}(R_{P,i} - R_P)(R_{Q,i} - R_Q)}{\sqrt{(R_{P,i} - R_P)^2}\sqrt{(R_{Q,i} - R_Q)^2}} \quad (1)$$

From the above equation (1), the similarity between programmers '$Prog_P$,' and '$Prog_Q$' is obtained based on the 'R' ranking given to attribute 'i' by programmer 'P' and from the summation fusion of the selected neighborhood programmer also. The resultant similarity value is a binary representation denoted as either '$+1$' or '-1' and is mathematically represented as given below.

$$S(Prog_P, Prog_Q) = \begin{cases} +1, Attributes\ are\ selected\ for\ further\ processing \\ -1, Attributes\ are\ eliminated\ due\ to\ redundancy \end{cases} \quad (2)$$

Prospects that are produced by Contingency Matrix Collaborative Filtering technique are either to be forecast or prospect. Forecast isa numerical value, 'F_{ij}', representing the ranking of attribute 'j' for the programmer 'i' and its resultant value represents either '$+1$' or '-1' based on the resultant Pearson Correlation Coefficient. On the other hand, Prospect represents the index of 'n' attributes that the programmer will like the most. The pseudo code representation of project attribute selection using Contingency Collaborative Filter is as given below.

Input: Programmer '$Prog_1, Prog_2, \ldots, Prog_n$', Project attributes '$Att_1, Att_2, \ldots, Att_n$',
Output: Pre-processed project attributes '$A_1, A_2, \ldots, A_n$'
1: **Begin** 2: **For** each Programmer '$Prog_1$' 3: Measure the similarities based on the neighborhood using equation (1) 4: **If** $S(Prog_P, Prog_Q) = +1$ 5: Project attribute selected 6: **End if** 7: **If**$S(Prog_P, Prog_Q) = -1$ 8: Project attribute not selected 9: **End if** 10: **End for** 12: **End**

Algorithm 1 Contingency Collaborative Filtering algorithm

As given in the above Contingency Collaborative Filtering algorithm, the problem of project attributes overload, which is a very customary occurrence with attribute retrieval (i.e. project attribute) is said to be solved. To solve this issue, the Contingency Collaborative Filtering algorithm uses similarities based on the Pearson Correlation Coefficient that not only implies either the presence or absence of correlation between two programmers for extracting the project attributes, but also obtains the exact extent to which the project attributes or two programmers are correlated. This in turn improves the project attribute selection accuracy and time.

3.2 Expectation Attribute Maximization Clustering

With the pre-processed project attributes, the second step in the proposed framework is the design of a clustering model, where the pre-processed attributes are divided into groups, called, clusters via Expectation Maximization. This step is otherwise called as the Expectation Attribute Maximization Clustering model. Therefore, before dividing the pre-processed attributes into groups, the maximum length is first identified for the pre-processed project attributes 'A_i' and is mathematically formulated as given below.

$$L_P^n = MAX\{L_{P1}, L_{P2}, \ldots., L_{Pn}\} \quad (3)$$

From the above equation (3), for each project '$P1$', '$P2$', …, 'Pn' or '$P \in \{P1, P2, \ldots., Pn\}$', the Expectation Maximization Clustering algorithm generates a series for attribute 'A_i', is mathematically formulated as given below.

$$A_i^1(P_1), A_i^2(P_1), \dots, A_i^n(P_1), \quad (4)$$
$$A_i^1(P_2), A_i^2(P_2), \dots, A_i^n(P_2) \quad (5)$$
$$A_i^1(P_n), A_i^2(P_n), \dots, A_i^n(P_n) \quad (6)$$

The Expectation Maximization Clustering algorithm then divides a group of project attributes into different groups, and each group is named asa cluster. Therefore, by efficient clustering similar project attributes are grouped into a software project. Figure 3 shows the example of Expectation Attribute Maximization Clustering for three projects.

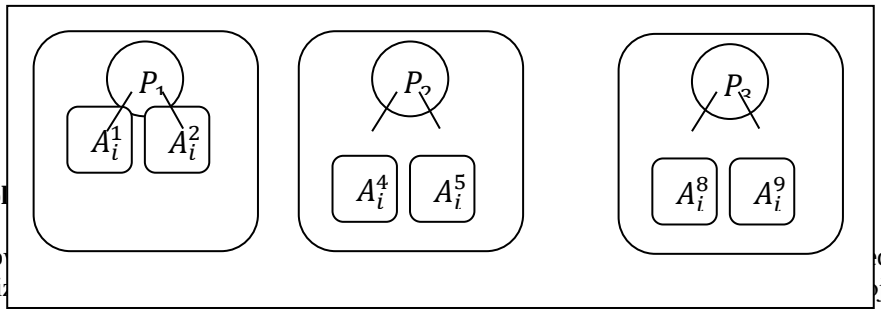

Figure 3 Sampl

As sho d, with which the expectation 'E' and maximi ject. It is given as below.

$$E = Prob\ (P \mid A, \theta) = Prob\ (P_1, P_2, P_3, \dots., P_n \mid A_i^1, A_i^2, \dots., A_i^n, \theta) \quad (7)$$
$$M = MAX\ L(\theta) = E_{Prob\ (P \mid A, \theta)} \left[Ln \left(Prob\ (P_1, P_2, P_3, \dots., P_n \mid A_i^1, A_i^2, \dots., A_i^n, \theta) \right) \right] \quad (8)$$

The pseudo code representation of Expectation Attribute Maximization Clustering is as given below.

Input: Pre-processed project attributes '$A_1, A_2, \dots, A_n$', Project '$P \in \{P1, P2, \dots., Pn\}$',
Output: observed prospect cluster '$Y_1, Y_2, \dots, Y_n$'
1: **Begin** 2: **For** each project 'P' 3: **For** each Pre-processed project attributes '$A_1, A_2, \dots, A_n$' 4: Identify the maximum length using equation (3) 5: Generate a series of attributes using equation (4), (5) and (6) 6: Measure the expectation value using equation (7) 7: Measure the maximization value using equation (8) 8: **End for** 9: **End for** 10: **End**

Algorithm 2 Expectation Attribute Maximization Clustering algorithm

As given in the above algorithm, with the objective of improving the clustering performance, for each pre-processed project attributes as input, the maximum length is initially identified, so that the series of attributes as generated for the identified maximum length. Followed by which the expectation and maximization values are derived with which form the referenceset. Specifically, foreach set, if the processed element is identical to the reference set element, then the particular attribute is included in the specific project or else excluded in the specific project. In this way only unique or similar attributes belonging to a particular group or project forms a cluster, minimizing the number of attribute values, therefore improving the clustering performance.

3.3 Longitudinal Regression Software Process model

Finally, a machine learningbased software process model is designed with the objective of assisting the project managers to arrive at a conclusion regarding which software process model are more suitable for a new project. This is performed in the proposed framework by designing Longitudinal Regression Software Process model.

Let the available prospect cluster be, '$Y_1, Y_2, \dots, Y_n$'. Along with each observed prospect cluster, 'Y_i', and let 'P_{ij}' denote the 'ith' observation of project 'P_j' and further assume with an error term 'ε', then the mathematical formulation is as given below.

$$Y_i = \alpha_0 + \alpha_1 P_{i1} + \alpha_2 P_{i2} + \cdots + \alpha_n P_{in} + \varepsilon_i \quad (9)$$
$$= \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} \left(Y_i - \alpha_0 - \sum_{j=1}^{n} \alpha_0 P_{ij} \right) \quad (10)$$

From the above equation (9) and (10), the proposed framework selects the least squares so that the sum of the squares of the errors 'ε_i^2' isminimized to analyze the changes of longitudinal data over time. The pseudo code representation of Longitudinal Regression for software process selection is as given below.

Input: observed prospect cluster '$Y_1, Y_2, \dots, Y_n$', project 'P', Pre-processed project attributes '$A_1, A_2, \dots, A_n$'
Output: Optimized software process selection
1: **Begin** 2: **For** each project 'P' 3: **For** each Pre-processed project attributes '$A_1, A_2, \dots, A_n$' 4: Measure longitudinal regression using equation (9) 5: Select MIN(ε_i^2) 6: **End for** 7: **End for** 8: **End**

Algorithm 3 Longitudinal Regression Software Process Selection

As given above, the Longitudinal Regression Software Process Selection helps in assisting the project managers to locate the optimal software process model. Different software process models normally levy several impacts on software processlevel constituents. Therefore, using Longitudinal Regression Software Process Selection, several optimal factors like project attributes, perform clustering with project attributes were comprehensively considered by project managers for a new project, and choose the mostappropriate software process model. This in turn had a positive impact of software process being selected and therefore the success rate.

4. Experimental setup

The following five open-source programs from http://sourceforge.net are used to ensure optimized selection of software process towards software maintainability. It included the following information provided in Table 1.

Table 1 Characteristics of software program code

S. No	Program	Description
1	faqforge	a tool for creating and managing documents
2	webchess	an online chess game
3	schoolmate	a solution for administering elementary, middle and high schools
4	time clock	a web-based time clock system
5	phpsysinfo	an utility for providing the system information like CPU time, memory and so on

The proposed framework is compared against the existing work such as Process Mining Extension to SCAMPI [1] and a Capability Maturity Model Integration-based Software Process Improvement (CMMI-based SPI) [2]. Experiment is conducted on factors such as project attribute selection time, project attribute clustering accuracy and success rate with respect to software program code.

5. Results analysis of CFMC-LR

The Collaborative Filtered Maximization Clustering and Longitudinal Regression (CFMC-LR) framework is compared against the existing Process Mining Extension to SCAMPI [1] and aCapability Maturity Model Integration-based Software Process Improvement (CMMI-based SPI) [2]. To conduct experiments, schoolmate from http://sourceforge.net is used. The SchoolMate includes the details regarding the elementary, middle and high schools.

It concentrates on four domains, namely, administration, teachers, students and parents. Here, the administration domain includes both the classes and users of the SchoolMate whereas the teachers'domain includes the details regarding assignments and grades. Information regarding students' domain are accessed through students and students'information can also be checked by the parents' domain. The experimental results using JAVA are compared and analyzed with the aid of graph form given below.

5.1 Impact of software project development cycle time

Software development cycle time refers the time with which the framework produces software with the highest quality and lowest cost in the shortest time. While selecting a software process, the most predominant factor to be considered is the project attributes. Not all the software project attributes are relevant. Hence, irrelevant project attributes and redundant project attributes have to be removed. Therefore, the software project development cycle time is highly influenced by the project attributes selected and is mathematically represented as given below.

$$PDCT = \sum_{i=1}^{n} Time\ (Att_i) * Size\ (SPC) \quad (11)$$

From the above equation (11), the software project development cycle time '$PDCT$', is obtained by the product of the time taken to extract relevant attributes '$Time\ (Att_i)$' and the size of the overall software program code '$Size\ (SPC)$'. Figure 4given below shows the software project development cycle time for selecting software process model over the administration domain of SchoolMate extracted from http://sourceforge.net.

The software program code size ranges from 500MB to 5000MB and the samples are provided as input using JAVA. From the figure, with an increase in the size of the software program code provided, the software project development cycle time also increases, though the curve is not observed to be exponentially same. However, when the size of software program code is between 500MB and 1500MB, the software project development cycle time shows a steady raise whereas, it increases when the size of software program code is between 2000MB and 3500MB and then increased when the sample provided was between 4000MB and 5000MB. This is because the software program code was extracted from different structures. As a result, the percentage increase or decrease in software project development cycle time is not exponentially same.

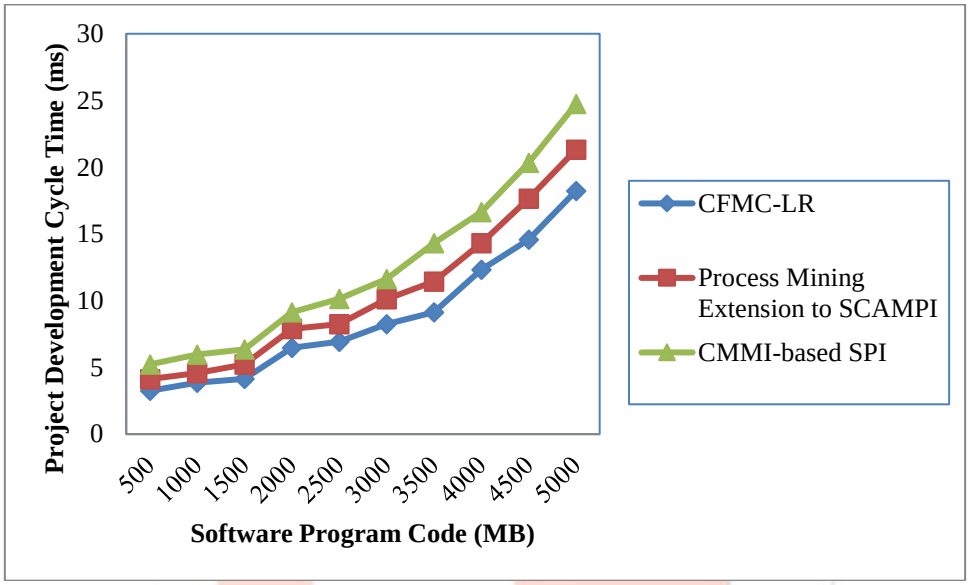

Figure 4 Impact of software project development cycle time

Figure 4 shows the impact of software project development cycle time for selecting software process and comparison is made with the existing methods, Process Mining Extension to SCAMPI [1] and aCapability Maturity Model Integration-based Software Process Improvement (CMMI-based SPI) [2]. From the figure it is clear that the software project development cycle time is comparatively lower when applied with the CFMC-LR framework than the two other existing methods.

The software project development cycle time is reduced using CFMC-LR framework due to the effective construction of contingency matrix using top 'n' list of attributes for programmer 'i' using collaborative filtering. The collaborative filtering in CFMC-LR framework effectively pre-processes the project attributes using contingency matrix based on the neighbourhood using Pearson Correlation Coefficient. Therefore, software project development cycle time using CFMC-LR framework is reduced by 18% and 31% compared to Process Mining Extension to SCAMPI [1] and CMMI-based SPI [2].

5.2 Impact of project clustering accuracy (reducing firm cost)

The software project clustering accuracy helps in identifying whether appropriate software process is identified in the early stage of development process. Higher the software project clustering accuracy, more the probability that the appropriate software process were selected. Hence, the project clustering accuracy is mathematically formulated as given below.

$$PCA = \left(\frac{No.of\ CRSP}{TSP}\right) * 100 \qquad (12)$$

From the above equation (12), the project clustering accuracy 'PCA', is the ratio of correctly recommended software process model '$No.of\ CRSP$' to the total software process model 'TSP' available in hand with the project managers. It is measured in terms of percentage (%). The convergence plot for software program code of size in the range of 500MB to 5000MB using teachers'domain that manages the students' assignments and grades are shown in Figure 5.

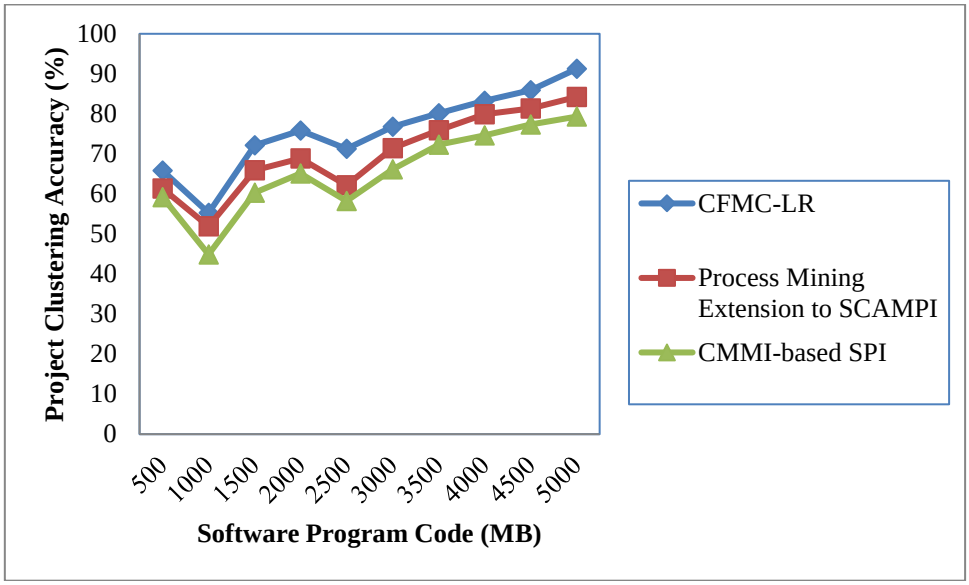

Figure 5 Impact of project clustering accuracy

From the figure it is illustrated that the proposed CFMC-LR framework achieved higher rate of clustering accuracy compared to other methods. It is also seen that, in Figure 5, the proposed Collaborative Filtered Maximization Clustering and Longitudinal Regression (CFMC-LR) shows a rise in the initial stature of the convergencegraphs during early iterations. However, when the size of the software program code was 1000MB, the rate of clustering accuracy decreases because of the extraction of data for each student differs in a varied manner.

The figure shows that the graph become tranquil and better project clustering accuracy is achieved with a drift increase when the size of software program code was 5000MB. The project clustering accuracy based on Expectation Attribute Maximization is improved by 8% when compared to Process Mining Extension to SCAMPI [1] due to the usage of pre-processed project attributes that develops higher ranked project attributes.Besides, applying Expectation Attribute Maximization Clustering algorithm, processed element are compared with the reference set element to form unique clustering. This in turn helps in providing unique list of similar attributes belonging to a specific project forming a cluster and therefore improving the project clustering accuracy by 16% compared to CMMI-based SPI [2]. Henceforth, with the improved accuracy, reduction in the cost of software process identification is said to be ensured.

5.3 Impact of success rate

In order to measure the efficiency of success rate obtained through the CFMC-LR framework, the software program code size and the source lines of code are considered during the experimentation.

Methods	**Success Rate (%)**
CFMC-LR	85.93
Process Mining Extension to SCAMPI	79.32
CMMI-based SPI	74.13

The software maintainability rate using CFMC-LR is provided in an elaborate manner in table 1 with different code size and experiment performed using JAVA.

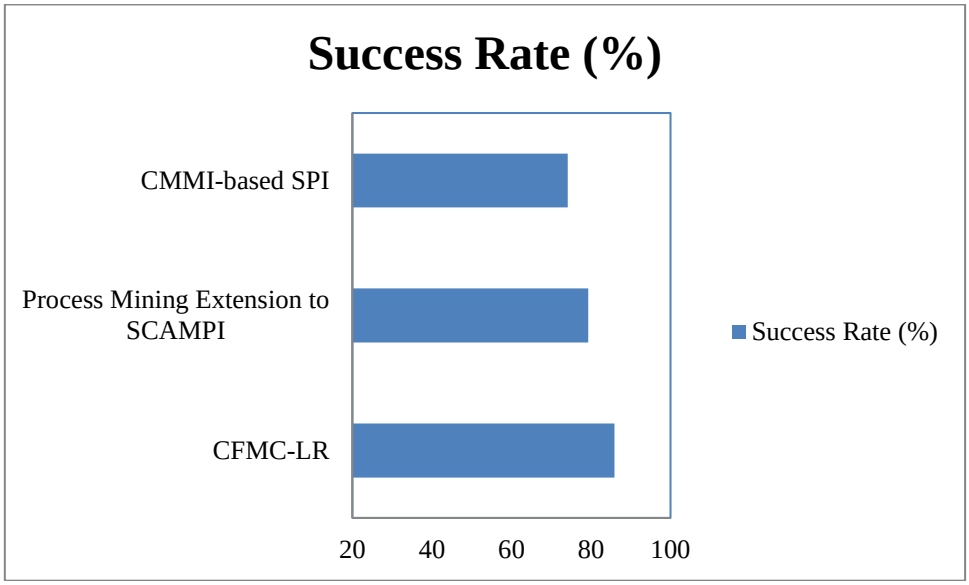

Figure 6 Impact of success rate

Figure 6shows the success rate for open source software code with respect to differing project code size and source lines of code at different time intervals. As depicted in the figure with the increase in the project code size, the success rate is also increased in all the three works. But when compared to the state-of-the-art works, the success rate is increased using the proposed framework CFMC-LR.

The success rate is improved using the proposed framework CFMC-LR owing to the fact that the proposed framework uses the Longitudinal Regression Software Process Selection algorithmthat selects the least squares so that the sum of the squares of the errors 'ε_i^2' isminimized. With the obtained intrinsic information, relevant project attributes are selected and used for clustering and accordingly, the software process is selected using Linear Regression. The advantage of using Linear Regression is that optimal software process model is selected that in turn decreases the cost of project life cycle and hence improving the success rate by 8% compared to Process Mining Extension to SCAMPI [1] and 7% compared to CMMI-based SPI [2] respectively.

6. Conclusion

Selection of software process by the project manager at an early stage remains a complex problem that attracts researchersto study and to try different methods and mechanisms to solve it. In this work, a new machine learning framework for selecting a software process is presented inorder to help project managers to take their decisions using Collaborative Filtered Maximization Clustering and Longitudinal Regression (CFMC-LR).A Contingency Matrix Collaborative Filtering techniquehas been designed towards software process selection for open source software code and therefore to improve the success rate of software process being selected at a faster pace. The CFMC-LR framework utilizes clustering model in the in the analysisprocess, and we have used expectation maximization in a bid to perform efficient clustering with the pre-processed project attributes. Next, with the efficient cluster being selected, based on the machine learning technique, appropriate software process has been selected by the project manager at an earlier stage of project development. To this, Linear Regression was applied to extract the appropriate software process model, therefore reducing the software development life cycle time. Experimental evaluation is conducted with the open-source program to measure the effectiveness of the proposed framework and parameter analysis are performed in terms of software development life cycle time, project clustering accuracy and success rate with respect to differing project code size and source program code. Compared to the existing software process withopen source software projects, the proposed CFMC-LR framework decreases the software development life cycle time andimprove the success rate by 49% and 15% respectively, whereas the project clustering accuracy is said to be improved by 24%.

References

[1] Arthur M. do Valle, Eduardo A.P. Santos,Eduardo de F.R. Loures, "Applying Process Mining Techniques in Software Process Appraisals", Information and Software Technology, Elsevier, Jan 2017 (Process Mining Extension to SCAMPI)

[2] Jung-Chieh Lee, Wen-Chin Hsu, Chung-Yang Chen, "Impact of absorptive capability on software process improvement and firm performance", Information Technology and Management, Springer, Dec 2016 – Capability Maturity Model Integration-based Software Process Improvement (CMMI-based SPI) [2]

[3] Ahmet Dikici, Oktay Turetken, Onur Demirors, "Factors Inßuencing the Understandability of Process Models: ASystematic Literature Review", Information and Software Technology, Elsevier, Oct 2016

[4] Nadja Damij, Pavle Boškoski, Marko Bohanec, Biljana Mileva Boshkoska, "Ranking of Business Process SimulationSoftware Tools with DEX/QQHierarchical Decision Model", PLOS ONE | DOI:10.1371/journal.pone.0148391 February 12, 2016

[5] Mujtaba Alshakhouri, Jim Buchan, Stephen G. MacDonell, "Synchronised visualisation of software process and product artefacts:Concept, design and prototype implementation", Information and Software Technology, Elsevier, Jan 2018

[6] Martin Fischbach, Dennis Wiebusch, and Marc Erich Latoschik, "Semantic Entity-Component State Management Techniquesto Enhance Software Quality for Multimodal VR-Systems", Transactions on Visualization and Computer Graphics, Mar 2016
[7] Patrick Rempe and Parick Mader, "Preventing Defects: The Impact of RequirementsTraceability Completeness on Software Quality", IEEETransactions on Software Engineering, Mar 2016
[8] George A. Sielis, Aimilia Tzanavari and George A. Papadopoulos, "ArchReco: a software tool to assistsoftware design based on context awarerecommendations of design patterns", Journal of Software Engineering Research and Development, May 2017
[9] Cuauhtemoc Lopez-Martõn, Ali Bou Nassif, Alain Abran, "A training process for improving the quality of software projectsdeveloped by a practitioner", The Journal of Systems & Software, Elsevier, May 2017
[10] Arash Khosravia, Ab Razak Che Hussina, Mehrbakhsh Nilashia, "Toward software quality enhancement by Customer KnowledgeManagement in software companies", Telematics and Informatics, Elsevier, May 2018
[10] Emilia Mendes & Pilar Rodriguez& Vitor Freitas Simon Baker & Mohamed Amine Atoui, "Towards improving decision making and estimatingthe value of decisions in value-based software engineering:the VALUE framework", Software Quality Journal, Springer, Mar 2017
[11] Marc-Oliver Gewaltig, Robert Cannon, "Current Practice in Software Development forComputational Neuroscience and How to Improve It", PLOS Computational Biology, Jan 2014
[12] Javed Iqbal, Rodina Binti Ahmad, Mohd Hairul Nizam Md Nasir, Mahmood Niazi, Shahaboddin Shamshirband, Muhammad Asim Noor, "Software SMEs' unofficial readiness for CMMI_-basedsoftware process improvement",Software Quality Journal, Springer, May 2015
[13] Andreas Brunnert, Helmut Krcmar, "Continuous Performance Evaluation and Capacity Planning UsingResource Profiles for Enterprise Applications", The Journal of Systems & Software, Elsevier, Jan 2015
[14] Kiana Rostami, Robert Heinrich, Axel Busch, and Ralf Reussner, "Architecture-based Change Impact Analysis inInformation Systems and Business Processes", IEEE International Conference on Software Architecture, Apr 2017
[15] Torgeir Dingsøyr & Nils Brede Moe &Tor Erlend Fægri Eva Amdahl Seim, "Exploring software development at the very large-scale:a revelatory case study and research agenda for agilemethod adaptation", Empirical Software Engineering, Springer, Jun 2017
[16] Alok Mishra, Deepti Mishra, "Software Project Management Tools: A Brief Comparative View", ACM SIGSOFT Software Engineering Notes, May 2013 Volume 38 Number 3
[17] Moses Kehinde Aregbesola, "Evaluation of Quantitative Process and Software Quality Management in the Nigerian Software-House", International Journal of Computer Applications (0975 – 8887) Volume 168 – No.1, June 2017
[18] Thamer Al-rousan, Bassam Al-Shargabi, "A New Maturity Model for the Implementation of Software ProcessImprovement in Web-Based Projects", Journal of Digital Information Management, Apr 2017
[19] Yolanda M Fernández-Ordóñez, Jesús Soria-Ruiz and Antonia Macedo-Cruz, "A Software Process Framework for Guiding the Construction Specificationof Geospatial Databases", Journal ofInformation Technology & Software Engineering, May 2017
[20] J. C. Furtado and S. R. B. Oliveira, "A Process Framework for the Software andRelated Services Acquisition Based on theCMMI-ACQ and the MPS.BR Acquisition Guide", IEEE LATIN AMERICA TRANSACTIONS, VOL 10, NO 6, DECEMBER 2012