International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 11 (2019) pp. 208-218

Received: 7 Jan, 2019; Accepted: 12 July, 2019; Published: 6 August, 2019

Reengineering Cost Estimation using Scrum Agile Methodology

Jaswinder Singh[1], Kanwalvir Singh Dhindsa[2] and Jaiteg Singh[3]

[1] Department of Computer Application, IK Gujral Punjab Technical University
Kapurthala, Punjab, India
Jaswinder_luthra@yahoo.co.in

[2] Department of Computer Science and Engineering, Baba Banda Singh Bahadur Engineering College
Fatehgarh Sahib, Punjab, India
Kanwalvir.singh@bbsbec.ac.in

[3] Department of Computer Applications, Chitkara University
Rajpura, Punjab, India
Jaiteg.singh@chitkara.edu.in

Abstract: **Estimating the budget for developing software is one of the prime tasks for software stakeholders. Good estimation increases the customer faith and goodwill for the development company. Many estimation techniques exist for estimating the cost of the software. Estimating reengineering projects are equally important. Researchers estimated cost of Reengineering using conventional algorithmic estimation methods. They also used classical software development approaches to perform reengineering. Conventional estimation methods are suitable in an environment where requirements are predefined and fixed. Practically, these methods can not fit in today's software development environment. We need more realistic approach to estimate. Since a decade, we have witnessed a change in the Software development approaches. Now software development process is more people centric and realistic for their stakeholders. This change in process is due to Agile. Agile methodology has gained the interest of both customers as well as developers. The main objective of this research is to estimate the cost of reengineering with consensus based estimation technique of Scrum development methodology. Agile Reengineering model is also proposed for estimation and performing reengineering. Thus the research is aimed to provide a model, which not only helps in performing the reengineering estimations but also guides how to perform reengineering. Scrum approach with sprint iteration of three weeks is used to perform reengineering. Chidamber and Kemerer (CK) metric is applied to determine the complexity metrics for various classes of the software. Reengineering is performed to make the project more maintainable by reducing the CK metric complexity. Various tools used in this work include CK java Metric tool (CKJM) ver-1.9 for calculation of CK metrics suit, IBM Rational Rose ver7.5 for Unified Modeling, Rapid Minor studio ver7.1 for determining the reengineering requirements of the software.**

Keywords**:** Software Engineering, Software Reengineering, Reengineering Cost estimation, Agile Scrum Methodology,

I. Introduction

Reengineering is aimed to improve the quality of the existing software [1]. It makes the system more maintainable and also extends the life expectancy [2]. It is essential to decide when to reengineer the system. Estimating cost and efforts are crucial to determine the feasibility of software development [3]. As we Estimate software development cost, similarly cost of reengineering should also be estimated.

Once the requirements for the reengineering of the system arise, it is required to estimate the cost needed to perform reengineering. The process of reengineering mainly includes three phases. The first phase is to analyze the existing system (Reverse Engineering), the second phase is to inculcate the required reengineering requirement (Transition) and last but not least, verify and validate the entire system (forward engineering).

Many popular algorithmic cost estimations methods exist to estimate the cost of software development like Function Point (FP), the Source line of Code (SLOC), Constructive Cost Model (COCOMO). Many authors like Sneed [4] and Sood [5] performed cost estimations for software reengineering process using conventional estimation methods.

The main problem with these existing methods is that they are not suitable in an environment where software requirements continuously keep changing. Traditional algorithmic methods are static. They worked well when software requirements are already known and fixed. So the need is to have a more realistic and flexible approach to determine the cost estimations for the reengineering process.
For accurate estimations, it is essential to know what methodology will be used to perform reengineering. Use of right approach will positively affect the estimation accuracy. In this research work, Scrum approach of agile is used to

perform reengineering of the software. Agile methodology already gained lots of importance in today's IT Industry [6][7]. Many Software development companies rely on Agile Methodology [8][9]. Agile is a fast and easy development methodology, and it also provides a quick response to change [10].Chaos Report [11] compare the success rate of Agile and Waterfall's approach for all size projects from the duration of 2009 to 2013. More than ten thousand projects were analyzed in this duration. The study by Chaos, measures the percentage of successful projects, challenged (Projects that missed time or cost or feature target) and failed projects. The analysis shows that agile methodology performed much better as compared to the waterfall approach. Comparison between waterfall and agile method for all sized project is given in Figure 1 below.

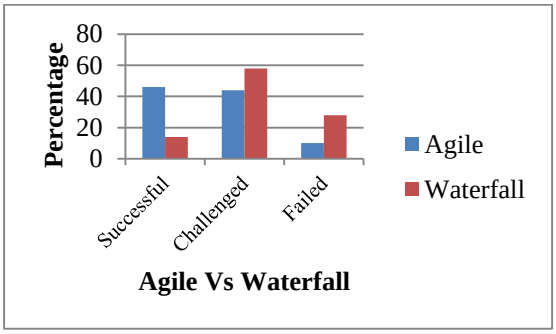

Figure 1. Agile Vs Waterfall Software Development [11]

Thus the proposed research work uses the advantages of agility for reengineering cost estimations and implementation.

II. Related Work

Contribution of various researchers in field of reengineering, agile methodology and cost estimation is covered in this section.

A. Reengineering cost estimation

As defined by Chikofsky et al. [12] "Reengineering is both renovation and reclamation, are the examination and alteration of the software system to reconstitute it in a new form and the subsequent implementation of the new form". Figure 2 shows General Reengineering model. Reengineering is performed using reverse, alteration, and forward engineering.

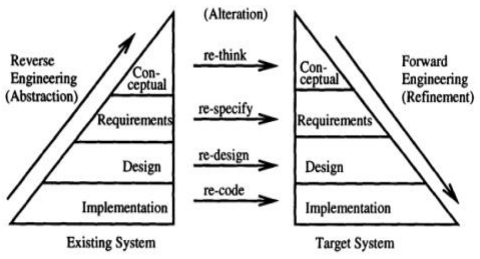

Figure 2. Reengineering Model [13]

Many researchers contributed in estimation of reengineering cost. Table 1 shows cost estimation work of various researchers.

Sr No	Author	Research contribution	Cost Estimation Method	Remarks
1	Sneed [4],[14]	Estimated the cost of reengineering using eight analysis steps. Efforts were calculated using various factors like adjusted system size, quality adjustment factors, adjusted productivity factor, risk adjustment factor. Time is calculated using the modified COCOMO model. In another research work, Sneed also proposed various reengineering parameters for cost analysis.	Size, COCOMO	Used well-known Cost estimation method but algorithmic estimation is good when software requirements are already known and fixed and no further changes in the requirements required.
2	Zou [15]	Estimated the efforts and cost of maintenance and reengineering using different estimation factors. Factors include the number of programs, input/output, subroutines, files, etc.	Size based Estimation	Estimation based on conventional Estimation Technique. The research considered various factors for an accurate estimate.
3	Tzerpos [16]	Research work Identified various cost/benefit factors for	Quality, Tool Support, Requirements	Proposed various qualitative factors.

		software reengineering based on quality, tool support, data conversion required and expert availability.	of data Conversion, Expert Availability	
4	Sood [17]	Estimated the reengineering cost using fault cost and defect cost.	Fault Cost, Defect cost	Based on Subjective measures to calculate values of reengineering cost.
5	Kumaw at et al. [18]	Determine the cost of reengineering using adjusted function point method. Compare the adjusted function point, use case and line of code methods	Adjusted Function Point (AFP)	Used unadjusted function point count and a value adjustment factor

Table 1. Earlier reengineering cost estimation work.

B. Agile Software development and Scrum

According to Abrahamsson et al. [19], development method is agile when "Software development is incremental, communication of customer and developers is close and consistent, Method itself is easy to learn and modify and able to make last moment changes." Agile methodology is very flexible in terms of adaptability in changing requirements. According to Ceschi et al. [20], companies using agile approach will have a better customer relationship, project planning satisfaction level, and change requirement adaptability as compared to plan based companies?

There are various Agile approaches [21][22] like Lean programming, Extreme Programming, United Process, Kanban, FDD (Feature-Driven Development), Scrum, Crystal, DSDM (Dynamic Systems Development Method). In this research work, Cost estimation and reengineering is performed using the Scrum approach of agile Methodology.

Scrum is one of the highly used approaches for agile by software companies [22]. In the 12th annual state of agile report, version-one declared that the scrum continues to be a popular choice of the developers and users of scrum methodology is 56% as compared to other agile methods [23]. Role of Scrum is Significant when there is a need to make changes in the existing projects [19].

Scrum uses iterative and incremental development as shown in Figure 3 below.

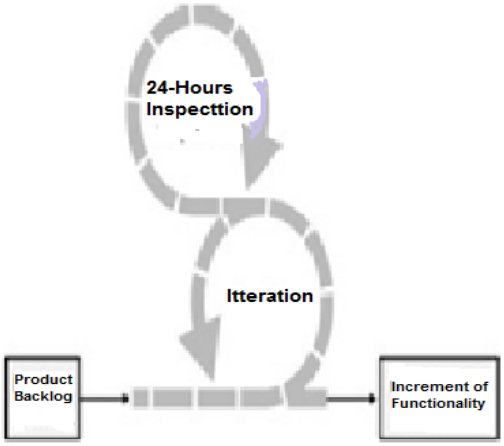

Figure 3. Scrum process [24]

In Scrum, user stories (Requirements) are collected by the development team in project backlog where these stories are prioritized and estimated, implemented in several iterations called sprints and incremented to achieve the final project. Each Sprint provides a working model to the user. Sprints are assigned with fixed timeframe, and the development team must implement the task or story by the end of the timeframe. The development team here does not mean programmer only. Tester, Programmers, Analyst, a Database administrator can be part of the development team. Daily 10-15 minutes Scrum meetings are the key for successful sprint implementations. Scrum approach is having six significant roles [19] identified as Scrum Master, Product Owner, Scrum Team, Customer, Management. Team starts the project with the first setting product backlog. User stories in the product backlog are identified with initial 2-3 hrs of interaction with a customer. Scrum process is shown in Figure 4. The whole process is divided into three parts, planning Phase, development phase and then post-development phase.

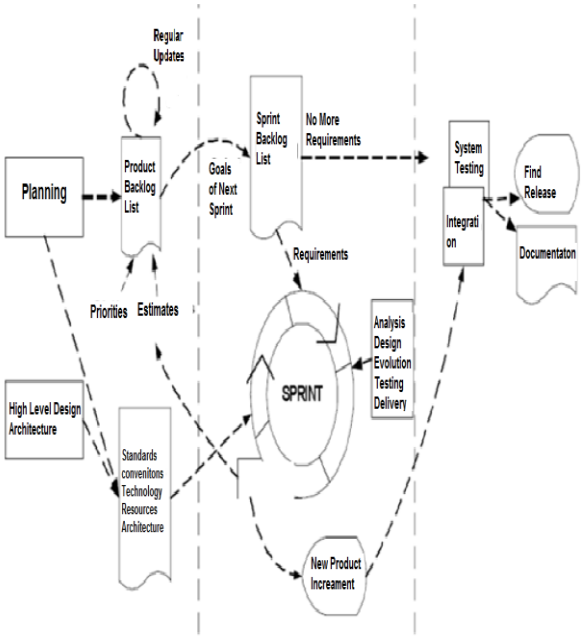

Figure 4. Scrum Process [19]

Story points are assigned to each user story using estimation technique.

C. Planning Poker in Scrum

Planning poker also called Scrum Poker is one of the trusted estimation technique used in the Scrum approach [25]. As stated [26] "Planning poker combines expert opinion, analogy, and disaggregation into an enjoyable approach to estimating that results in quick but reliable estimates. Participants in planning poker include all of the developers on the team." Planning Poker is based upon the old estimation technique named as Delphi. In this technique, estimations were based upon the elicitation and judgment. Usman et al. [27] have identified planning poker as one of the most useful technique of effort estimation in agile software development. Planning poker is empirically used for estimation of user story and task in the work of Haugen [28], Molokken-ostvold, et al. [29] and Mahnic et al. [30].

1) Planning poker benefits includes

- Scrum team sit together to perform estimations. Team includes scrum master, development team (developer, Analyst, testers) and Product owner.
- In the meeting, team discussed both high and low estimation and decision is made after proper discussion. It avoids the problem of conflict for the future.
- Work begin when all Scrum team members are agreed upon the same consensus so commitment for the projects increase
- Dominance or biasing is avoided as everyone gets a chance to justify himself and everyone‘s opinion is welcomed.

2) Estimation process

Planning Poker includes the following steps to estimate user stories.

- Before starts of each sprint, Scrum Master Schedule session with the team to play planning poker with poker cards (say with card number 1 2 3 5 8 and so on).
- Requirements are projected and understood,
- Each team member open a deck of cards and reveal his estimate
- If each team member reached the same consensus, then estimations for the next requirement starts
- Otherwise lower and higher estimator gives a justification, and the requirement can be estimated again
- Once all requirements estimated then team begins with the development of sprint.

The estimation method can be understood with a simple example. Suppose team consist of Scrum Master named Mike and three Team members named Alin, Jone, and Smith. Poker cards are carrying numbers 1, 2, 3, 5, 8, 13, 20 and so on. Estimation is performed for the following scenario.

User story: In the e-commerce web site, if Customer searches any item then recent similar purchase items should also appear to make the customer better choice in the purchase.

User Story	Alin	Jone	Smith
In the e-commerce web site, if any customer searches any item, then recent similar purchase items should also appear to make the customer better choice in the purchase.	3	5	8

Table 2. Planning Poker Result

After playing the Card, the estimates by three people are shown in Table 3. All have different estimations Alin gave estimation as three poker points, the lower one and Smith gave as eight poker point, the higher one. A time-bound discussion will get started. Alin justifies that he already worked on a similar scenario and know the algorithm. Smith view is that the algorithm is useful but it requires lots of parameters, and an updated version of the algorithm is also available. At this point if Alin can convince all members and all members are convinced then the user story is estimated using three-story point otherwise re-estimation takes place. After the re-estimation, the result is given in Table 3 below. At this point, Mike may ask Jone to share his view. As per john, a new algorithm can be used with little uncertainty as he as an experience of using a new version with different parameters.

User Story	Alin	Jone	Smith
In the e-commerce web site, if any customer searches any item, then recent similar purchase items should also appear to make the customer better choice in the purchase.	5	5	8

Table 3. Planning Poker Re-estimation Result

Now Mike can assign five as story point with the consensus of all as majority number. Re-estimations can further carry on if the lower and upper estimation by member differs by two steps (3 and 8 or 5 and 13) or team is not convinced upon the opinion.

In this research, planning poker method is used to estimate the Cost of the reengineering process.

D. Integrating Reengineering with Agile software development

Many Researchers have integrated reengineering and agile development approach.

Sahoo et al. [31] integrated two approaches in N-process model. In this model, iterative reengineering process is defined in three phases named the reverse engineering leg, the reincarnation leg, and the validation leg. Mainly focused on reverse engineering is shown using implementation

sequence diagram (ISD) and implementation class diagram (ICD). Chung et al. [32] give an idea of service-oriented software reengineering using Scrum. They Ran multiple scrums in parallel with separate scrum teams for reverse and forward engineering. Multiple scrums in parallel used to provide visual models for service consumption and service production. Cagnin et al. [33] represent a framework called PARFAIT which used GRN pattern language and rational unified process phases (Inception, Elaboration, Construction, and Transition phases) to provide prototype at the very initial stage of reengineering process. Adrian [34] used a short, agile iteration process in the area of aspect-oriented reengineering. The author used various refactoring iterations to analyze the effect on the modularity of the project.

III. Proposed Agile Reengineering Model

Proposed work is to measure reengineering cost estimation using planning poker estimation in agile Scrum methodology.

Figure 5 shows integration of reengineering tasks with agile scrum methodology. Proposed model retains the essence of agile and reengineering approach. Three phases of reengineering are implemented using Agile Scrum methodology. Sprint of three-week iteration enclosed all three reengineering tasks that is reverse, alteration and forward engineering.

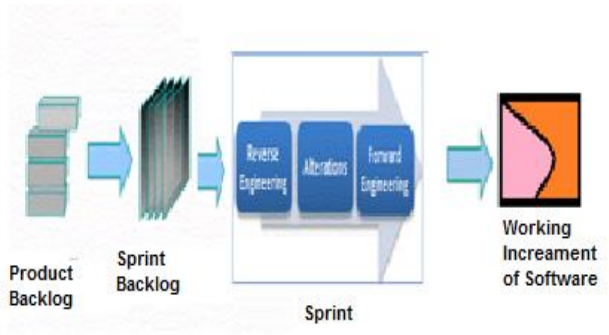

Figure 5. Proposed Agile Reengineering Model

In proposed work, one sprint performs all the reengineering tasks that are Reverse, alteration, and forward engineering. The reengineering tasks might take many numbers of sprints. Completion of each sprint is integrated with the software, and regression testing is performed to ensure successful integration.

Agile reengineering model works as follows.

- Initial iteration planning (time allocation for iteration, team required, and other resources required) is performed in Scrum meeting. Planning is done including all stakeholders.
- Planning includes number of weeks per iteration (Sprint) to accommodate reverse, forward and alteration.
- Reengineering Requirements analysis in terms of estimating user stories, allocation of story points is performed using planning poker.
- Estimation of cost of reengineering.
- Main reengineering requirement is to reduce complexity of modules to make the software more maintainable.
- Requirements required to implement in one sprint are assigned to the sprint backlog.
- One sprint is used to perform the tasks of forward, alterations and reverse engineering for the requirements.
- Daily planning in short scrum meetings is performed every day.
- Retrospective actions are taken to ensure the implementations of required objectives. Re-estimation requirements and speed of requirement implementations (velocity) are also verified.
- All increments are integrated to get the final system and tested to ensure error free integration.

A. Cost Estimation

Usman et al. [35] proved that planning poker is the most extensively used technique in agile software development for cost estimation as compared to the other existing estimation techniques and story point is the metric used most frequently to determine size. It is proved that planning poker estimations are more accurate when the team is having related experience with similar projects.

Molokken-ostvold et al. [29] and Mahnic et al. [30] also identified that planning poker produces more accurate estimations as compared to the statistical combination of expert's estimates. In this research work, planning poker in Scrum is used for reengineering cost estimations. User stories are estimated using planning poker. For large and complicated stories, it is better to break the story into the sub-tasks and similarly perform task estimations. Generally, User story points are assigned in the product backlog, and the task is estimated in the sprint backlog.

Story points and number of days required to accomplish the tasks are correlated with each other. As suggested by Cohn [26], one story point equals to one day. For example for 13 story points assigned to a task and if one point takes 12 hrs to accomplish then 13*12 is 156 hrs. Considering 6 hrs as actual efficient work by team member then 156*2 is 312 hrs (i.e. time taken to complete 13 point story). Total days would be 312/24 that is 13 days. Hence the whole time of iteration for 13 story point is 13 days. If we have to assign the iteration length based on this, then it will better to assign as iteration cycle of 3 weeks as the extended days can give developer to review and retrospective the development. As reengineering project undergoes through three different phases (Reverse, Alteration, and forward engineering), duration of iteration cycle must take care of complete fulfillment of all these phases. Selection of iteration length is essential. Various factors can affect this choice like project release time, case of uncertainty in work, stability in prioritizations, and ease of stakeholder feedback. The iteration length should neither be too long nor too short. It must give ample time to the team to do the work efficiently and accurately. It is also essential to prioritize user stories.

Thus in this proposed work, one Sprint of 3-weeks duration completes the entire reengineering process.

As measured by Cohn [26] estimation includes the workforce cost as well as hidden expenses that are nearly 50% of employee annual salary. Accumulated Salary includes employee's annual Salary and all other expenditure for resources provided to employee. Accumulated salary per iteration represents expenditure per iteration. Cost estimations for the reengineering project can be computed using the following formulations.

Accumulated Salary expenditure
= Annual salary + 50% * (Annual salary due to the expenditure of another source) (1)
Accumulated Salary per iteration
= Accumulated Salary / (52/Total time of one iteration that is three weeks) (2)
Accumulated Cost per time spent
= (Accumulated Salary per iteration) * (percentage of total time spent per iteration). (3)
Total cost per iteration
= Sum of Accumulated Cost per time spent (4)
If x number of story points per iteration then
Cost of story point per iteration is
= (Total cost per iteration) / x. (5)

IV. Validation and Analysis of re-engineering cost estimates

Cost estimations are made for those projects which required to be reengineered. Software is developed using Java named 'Code Level Security'. By applying the Chidamber and Kemerer (CK) metric suit [36] as shown in Table 4, Complexity metrics for each module is determined. Six basic metric set of CK metric suit used in this paper includes Weighted Methods per Class (WMC), Coupling Between Object classes (CBO), Depth of the Inheritance Tree (DIT), Number of Children (NOC), Response for a Class (RFC) , Lack of Cohesion of Methods (LCOM).

Sr No	Metric & Classes	WMC	DIT	NOC	CBO	RFC	LCOM
1	Login	12	6	0	9	78	60
2	IDE	17	6	0	17	121	70
3	User detail	23	5	0	12	109	183
4	program access report	12	6	0	8	89	62
5	Profile detail'	11	5	0	6	74	25
6	User report	8	6	0	6	77	24
7	Saved program report	14	6	0	10	98	73
8	User maintenance	2	1	0	1	5	1
9	Program update report	12	6	0	9	94	48
10	Main frame	22	6	0	19	89	233
11	program report	8	6	0	6	74	24

Table 4. CK metric Suit Complexity measure [37]

Reengineering requirements for the projects are listed in Table 5. The requirements need to be estimated and prioritized. As discussed above, there can be various factors that can influence this estimation like team member's prior experiences, skill set, level of customer involvement, etc. Size estimations for user story are made using planning poker. Story points are determined and assigned to each story.

Sr No	Sprint Backlog	Estimated Poker Points
1	The complexity of "user detail module" should be reduced. It should be more maintainable.	5
2	The complexity of "IDE module" should be reduced. It should be more maintainable	8
3	The complexity of "Login module" should be reduced. It should be more maintainable	2

Table 5. User Story estimations for sprint backlog

For performing reengineering, a sprint of 3-week iteration length is decided. This cycle will include reverse engineering, alteration, and forward engineering. Estimation using story point is called as a relative estimation. User stories are further divided into the task with an hour as a unit. It is also called as an absolute estimate. The tasks are shown in Table 6 below.

Sr No	Tasks	Hours
1. The task of Reverse Engineering		
1.1	Re-documentation /document generation Recovery of design	6
1.2	Analysis of High-level design	4
1.3	Analysis of Low-level design	8

1.4	Analysis of components that require changes or restructuring	12
2. The task of Alteration and Forward Engineering		
2.1	Remodeling the behavior of classes	6
2.2	Changes in the classes to reduces the design complexities	18
2.3	Unit testing	6
2.4	Regression testing	12
2.5	Integration with the existing modules	6
2.6	Integration testing	6
2.7	Retrospective	6
	Total Sprint Time	90 Hrs

Table 6. Sprint Tasks

The cost of the project is estimated as shown in Table 7. By using planning poker, we have estimated user stories and performed hourly distribution of takes as shown in Table 5 and Table 6 respectively. Now we can easily determine the Cost of one sprint or total cost per iterations using equations discussed in section III. All formulation equations (1), (2), (3) and (4) are measured and results are summarized in Table 7.

Scrum Team	**Salary/ year in Rs.**	**Accumulated Salary**	**Accumulated Salary/ iteration**	**Total time spent**	**Accumulated Cost per time spent**
Scrum Master	60000	90000	5200	100 %	5200
Programmer	50000	75000	4350	100 %	4350
Tester	50000	75000	4350	100 %	4350
Total Cost per iteration					13900

Table 7. Cost Estimations

Thus using table 7, measuring cost of story point per iteration is 13900/15 that is 927 approximately. For three iterations, the total cost of the project will be 13900*3 that is 41700. Thus cost estimation determines the total cost of the project.

As measured in the table 7, proposed estimations are easy to measure and any change in the value of data can easily be applied in given formulations. Suppose for new project, time spent by tester is 75% then the accumulated cost per time spend by the tester will be 4350*.75 that is 3262 Approximate. Not only the cost, but proposed work can also determine the schedule of the project. If the velocity of the task performed by team is 15 story points per 3-week iterations and total story points get estimated is 260 then total iterations required will be 260/15 that is nearly 17 iterations. Total project duration will be 17*3 that is 51 numbers of weeks.

Although cost is empirically calculated there are always uncertainty factors that can affect the estimations. In agile development, Cohan [26] suggested uncertainty about +-25% in the estimates.

Thus as compared to reengineering cost estimation used by various researchers as given in Table 1, proposed cost estimation has following benefits

- Estimations are consensus based and involved all stakeholders
- Less time to estimate and more accuracy as compared to conventional cost estimation methods of reengineering.
- Allow Re-estimation if there is any change in requirements.

Once reengineering cost is estimated, reengineering using the proposed Agile Reengineering model is also performed in the next section.

V. Reengineering Performed

After estimation of cost of reengineering project, reengineering is applied to the modules of the software using Scrum. Reengineering is performed in one sprint of 3-weeks duration.

After reengineering the project, the CK metric values for IDE, Login and user details have been reduced to a greater extent.

Sr No.	**Class & Metric**	**Login**		**IDE**		**UserDetail**	
		NEW	**OLD**	**NEW**	**OLD**	**NEW**	**OLD**
1	WMC	4	12	4	17	6	23
2	DIT	6	6	6	6	6	5
3	NOC	0	0	0	0	0	0
4	CBO	5	9	12	17	4	12
5	RFC	4	78	10	12	6	109
		8		2	1	7	
6	LOCM	2	60	0	60	0	183
	Total	6	16	12	22	8	332
		5	5	4	1	3	

Table 8. CK metric comparison before and after reengineering

As shown in Table 8, there is a drastic reduction in the lack of cohesion for the reengineered project. Weighted method per class (WMC), the coupling between projects (CBO), response for class metrics (RFC) are also reduced in the reengineered project. The reasons for these changes are the reengineering tasks performed on the project. These changes are summarized in Table 9 below

	Reengineering Tasks
1.	Reverse Reengineering
1.1	Static structured is analyzed using UML class diagrams, and dynamic behavioral is understood using interaction diagrams
1.2	Data sets are explained on the local machine and

	attributes are created on a local computer to understand the working of the project.
2.	Alterations & forward engineering
2.1	Login, IDE, User detail forms were entirely re-build with the same functionality as the previous one. "actionPerformed" event on the form was bound and performance against CK metric is analyzed.
2.2	With the team discussion, forms were again rebuilt using an interface "ActionListener" and the following changes have been made • "ActionListener" was implemented in the class. • All methods inside that interface were overridden within the class. • A method name "ActionPerformed" was the key we needed. All the eight buttons namely (Prev, Next, First, Last, New, update, Save, Cancel) now jump to single function only. • A function named "getActionCommand()" was used to identify which button has been clicked. • Switch/case is applied for the events according to button clicked and wrote the coding as per our needs.
2.3	Unit testing for each module is performed, and the integration of modules is done in the overall system
2.4	Regression testing is performed to see the effect of rebuild forms in overall project execution by applying various test cases.
2.5	performance against CK metric is analyzed, and reduction in the complexity is noted

Table 9. Reengineering Process Tasks

The difference between the earlier and reengineered system are analyzed pictorial by using Unified Modeling Language class diagrams.

Changes made in data as well as the functions are visualized through class diagrams for reengineered modules.

A. Static Modules Representation

Changes performed in modules are depicted using UML class diagrams. The class representation shows the module before and after modifications.

1) UserDetail [Before Reengineering]

Figure 6 shows the static visualization of UserDetail Class before reengineering. It contains various functions to perform the tasks.

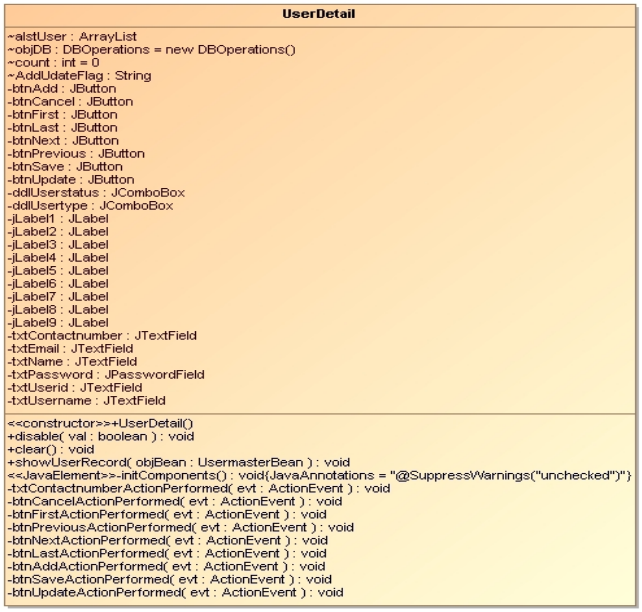

Figure 6. UserDetail Class before reengineering

2) UserDetail [After Reengineering, Named ActionListenerFrame]

Figure 7 shows the static visualization of UserDetail Class after the reengineering. The class shows the actionPerformed method bound on the form. Many different functions are encapsulated under the actionPerformed function.

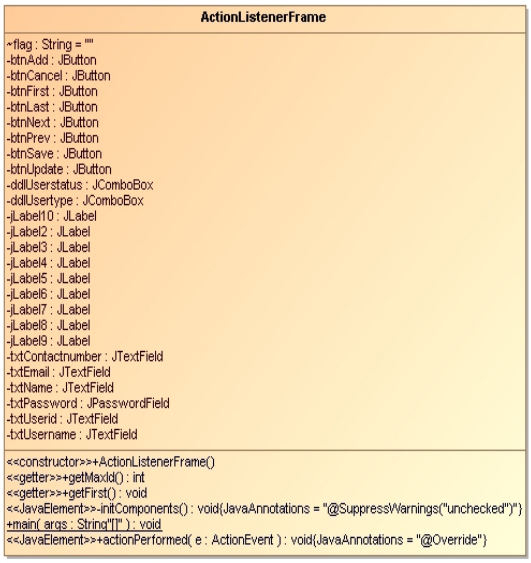

Figure 7. UserDetail class after reengineering

3) Login[Before Reengineering]

Figure 8 shows the static visualization of Login Class before reengineering. It contains various function defined as Login, Password, Cancel, Retrieves Password.

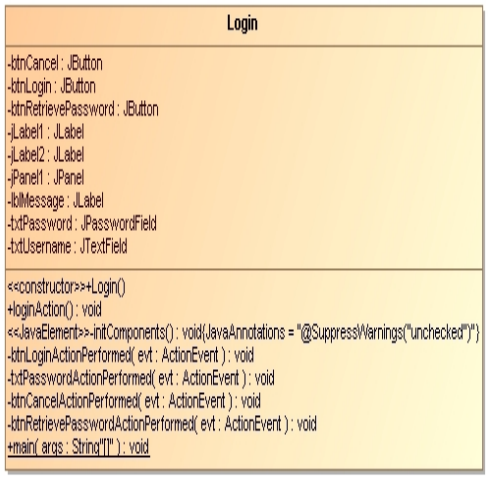

Figure 8. Login class before reengineering

4) Login Class [After Reengineering]

As shown in Figure 9, in this class the various function defined as Login, Password, Cancel and Retrieve Password are encapsulated in the actionPerformed method.

Figure 9. Login Class after reengineering

5) IDE [Before Reengineering]

Figure 10 shows the static visualization of IDE Class before reengineering. Multiple actionPerformed events can be seen in the Class Diagram.

Figure 10. IDE Class before reengineering

6) IDE [After Reengineering]

Figure 11 shows the static visualization of IDE Class after reengineering. Multiple events are included in actionPerformed method in the Class Diagram.

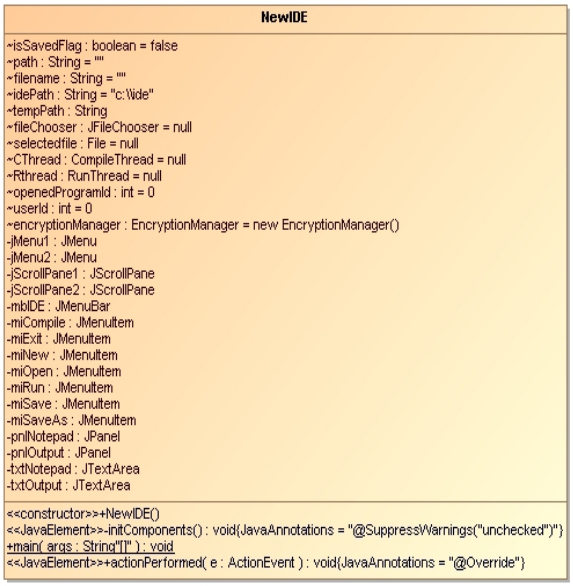

Figure 11. IDE Class after reengineering

B. Maintainability Improvement

Maintainability is a crucial quality factor in software engineering and predicting the maintainability is critical. Various researchers have proposed maintainability models and methods time to time. For object-oriented software maintainability, one of the useful models is suggested by Li et al. [38]. In their proposed model, use of CK metric suit is very prominent. Their work validated that CK metrics like DIT, NOC, RFC, LCOM, and WMC contributes to the prediction of maintenance efforts and by reducing the values of CK metric maintainability can be improved. As the results are shown in Table 8, reengineering efforts in reduced complexity of modules to a significant level. The maintainability improvement for the modules of classes is measured in Table 10.

Sr. No	Module	Maintainability Improvement	Percentage Maintainability Improvement
1	Login	165 to 65	60.6%
2	IDE	221 to 124	43.8%
3	UserDetail	332 to 83	75%

Table 10. Maintainability improvement in Reengineered modules

Figure 12 depicts the Percentage improvement in maintainability for all three modules after performing reengineering.

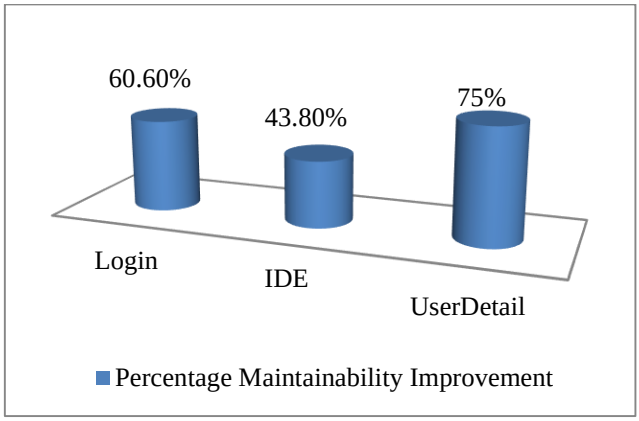

Figure 12. Maintainability Comparisons

Research work thus estimate reengineering cost and implement reengineering process using Scrum methodology.

VI. Conclusion

Proposed agile reengineering model estimates the cost of reengineering and performs reengineering implementation. Scrum approach of agile software development is integrated with the reengineering process. Estimation Cost method is easy to determine and more realistic. Estimations are performed using consensus based planning poker method of Scrum. Three weeks sprint cycle is used to implement the reengineering process. Cost Estimations are performed by assigning story points to the modules of the software using planning poker. The cost of story point per sprint, cost of one sprint and the total cost of the software is estimated for performing reengineering. In this research, CK metric suit measures the design complexity of the software modules. Three Modules having higher design complexities are selected for performing reengineering. Reengineering process is performed in three weeks Sprint. After performing reengineering, the reduction of Complexity of three modules is found at a significant level. Results also identified an increase in the maintainability of the reengineered Modules. The proposed research can be helpful to the software development companies which are using Scrum development approach to easily determine cost of reengineering. More modules of the software can be considered for reengineering using Scrum approach and an overall improvement in the software maintenance can be attained. An important point to be considered is that the team having good experience of a similar project will result in good estimation.

Acknowledgment

Thanks to IK Gujral Punjab Technical University for helping in accomplishment of this research work.

References

[1] H.M. Sneed. “20 Years of Software-Reengineering: A Resume”. In *Proceeding of the 10th Workshop on software reengineering (WSR’08)*, pp. 115-124, 2008.

[2] U. Kuhlmann. “Maintenance Activities in Software Process Models: Theory and Case Study Practice”. Institute of Software Engineering, University of Koblenz Landau, 2003.

[3] C. E. L. Peixoto, J. L. N. Audy, R. Prikladnicki. “The Importance of the Use of an Estimation Process”. In *Proceedings of the 2010 the ICSE Workshop on Software Development Governance*, pp 13-17, 2010.

[4] H.M. Sneed. “Estimating the Costs of a Reengineering Project". In *Proceeding of the 12th Working Conference on Reverse Eng*ineering, *IEEE CS Press*, pp. 111–119.2005

[5] S. Sood. “Software reengineering-A metric set based approach”, Ph.D. Thesis, Himachal Pradesh University, Himachal, India, 2012. http://hdl.handle.net/10603/129186.

[6] P. Serrador, J. K. Pinto. “Does agile work? - A quantitative analysis of agile project success”, *International Journal of Project Management*, *33*(5), pp.1040-1051, 2015.

[7] J. kisielnicki, A.M. Misiak. “Effectiveness of agile compared to waterfall implementation methods in it projects: analysis based on business intelligence projects”, *Foundation of management,* IX (1), pp. 273–286, 2017.

[8] J. Jeremiah, “Survey: Is agile the new norm?” Available at: https://techbeacon.com/survey-agile-new-norm, Published on May 25, 2015.

[9] P. Paganini. “Companies Worldwide Are Adopting Agile Development Techniques”. Available at: https://www.veracode.com/blog/2015/03/companies-worldwide-are-adopting-agile-development-techniques-sw, Published on April 1, 2015.

[10] Report on agile methodology available at: http://istqbexamcertification.com/what-is-agile-methodology-examples-when-to-use-it-advantages-and-disadvantages/ Accessed on June 20, 2018.

[11] Chaos report. Available at: :https://www.standishgroup.com/sample_research_files/CHAOSReport2014.pdf. Accessed on June 20, 2018.

[12] E. J. Chikofsky, J. H. Cross. “Reverse Engineering and Design Recovery: A Taxonomy”, *IEEE Software*, VII (1), pp. 13 – 17, 1990.

[13] E. J. Byrne. “A Conceptual Foundation for Software Re-engineering”. In *proceeding of the conference on Software Maintenance, 1992*, Orlando, FL, USA, pp. 226-235, 1992.

[14] H. M. Sneed, "Planning the reengineering of legacy systems", *IEEE Software*, XII (1), pp. 24-34, 1995.

[15] Y. Zou, “Software Reengineering (Reengineering Economics)”, Department of Electrical & Computer Engineering, Queen’s University. 2007

[16] V. Tzerpos. “Software Re-engineering”, Available at https://www.eecs.yorku.ca/course_archive/2005-06/W/6431/lec1.pdf

[17] S. Sood. “A Framework for Software Reengineering Using Set of Software Metrics”, *International Journal of Computer Science Engineering*, III (2), pp. 88-94, 2014.

[18] P. Kumawat, N. Sharma. “Design and Development of Cost Measurement Mechanism for Re-Engineering Project Using Function Point Analysis”, in *International Conference on Advanced Computing Networking and Informatics*, R. Kamal, M. Henshaw and P. Nair (eds.) , Advances in Intelligent Systems and Computing, Vol. 870. Springer, Singapore, 2019.

[19] P. Abrahamsson, O. Salo, J. Ronkainen, J. Warsta. *Agile Software Development Methods: Review and Analysis*, VTT, 2002. www.inf.vtt.fi/pdf/publications/2002/P478.pdf.

[20] M. Ceschi, A. Sillitti, G. Succi, S. De Panfilis. "Project management in plan-based and agile companies", *IEEE Software*, XXII (3), pp. 21-27, 2005.

[21] S.W. Ambler, M. Holitza. *Agile for Dummies*, IBM Limited Edition, John Wiley & Sons Inc. 2012.

[22] G. S. Matharu, A. Mishra, H. Singh, P. Upadhyay. “Empirical Study of Agile Software Development Methodologies: A Comparative Analysis”, *ACM SIGSOFT Software Engineering*, XL (1), pp. 1-6, 2015.

[23] Versionone. 12th Annual State of Agile Report. (2017) .Available At: https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report Accessed on July 3, 2018.

[24] K. Schwaber. *Agile project Management with Scrum*, Microsoft Press, 2004.

[25] M. Cohan, “Planning Poker”. Available at https://www.mountaingoatsoftware.com/agile/planning-poker, Accessed on July 3, 2018

[26] M. Cohan. *Agile estimating and planning*. Pearson Education, 2006.

[27] D. Usman, E. Mendes, F. Weidt, R. Britto. “Effort Estimation in Agile Software Development: A Systematic Literature Review”. In *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, pp. 82-91, 2014.

[28] N.C. Haugen. “An empirical study of using planning poker for user story estimation”. In *Proceedings of the AGILE 2006 Conference (AGILE’06), Minneapolis*, pp. 23–34, 2006.

[29] K. Molokkenostvold, N.C. Haugen, H.C. Benestad. “Using planning poker for combining expert estimates in software projects”, *Journal of Systems and Software*, pp. 2106–2117, 2008.

[30] V. Mahnic, T. Hovelja. “On using planning poker for estimating user stories”, *Journal of Systems and Software*, LXXXV (9), pp. 2086-2095, 2012. http://dx.doi.org/10.1016/j.jss.2012.04.005

[31] A. Sahoo, D. Kung, S. Gupta. “An Agile Methodology for Reengineering Object-Oriented Software”. In *proceeding of the 28th International Conference on Software Engineering & Knowledge Engineering (SEKE*), pp.227-232, 2016.

[32] S. Chung, D. H. Won, S. H. Baeg, S. Park, "A Model-Driven Scrum Process for Service-Oriented Software Reengineering: mScrum4SOSR". In *proceeding of the 2nd International Conference on Computer Science and its Applications*, pp. 1-8, 2009.

[33] M.I. Cagnin, J.C. Maldonado, R.D. Penteado, "PARFAIT: towards a framework-based agile reengineering process". In *Proceedings of the Agile Development Conference (ADC)*, pp. 22-31, 2003.

[34] O. Adrian. “Aspect-Oriented Reengineering of an Object-oriented Library in a Short Iteration Agile Process”, *Informatica*, XXXV, pp.499-511, 2011.

[35] M. Usman, E. Mendes, J. Borstler. “Effort Estimation in Agile Software Development: A Survey on the State of the Practice”. *In proceeding of the 19th international conference on evaluation and assessment in software engineering (EASE’15)*, pp.27-29, 2015.

[36] S. R. Chidamber, C. F. Kemerer, "A metrics suite for object oriented design", *IEEE Transactions on Software Engineering*, XX(6), pp. 476-493, 1994.

[37] J. Singh, A. Gupta, J. Singh. "Identification of requirements of software reengineering for JAVA projects". *In proceeding of the International Conference on Computing, Communication and Automation (ICCCA)*, pp. 931-934. 2017.

[38] W. Li and S. Henry, “Object-Oriented Metrics that Predict Maintainability”, *Journal of Systems and Software*, XXIII (2), pp.111-122, 1993.

Author Biographies

Jaswinder Singh holds a Masters in Computer Application and pursuing PhD in computer application. He is having 11 years of experience in Academics. Area of expertise includes Software Maintenance, Reengineering, Algorithm generations, and Data Mining. Current work includes Identification of Metrics for reengineering Java projects, Study of the significance of reengineering in Today's scenario for the Software development industry.

Dr. Kanwalvir Singh hold PhD in computer engineering .He is currently working as professor in the Department of CSE at Baba Banda Singh Bahadur Engg. College, Fatehgarh Sahib (Punjab). He has been awarded with the best PhD thesis award in international conference held in association with Computer Society of India. He has authored more than 90 publications in various esteemed international referred journals & proceedings of reputed international conferences. His area of interest includes Cloud Computing, Big Data, IoT, Mobile Computing, Database & Security, Web Engineering and software engineering..

Dr. Jaiteg holds a Ph.D. in Computer Science and Engineering with 12 years of experience in Research, Development, Training, Academics at Institutes of Higher Technical Education. His areas of expertise are Software Engineering, Business Intelligence, Data and Opinion mining, Cartography, Curriculum design, Pedagogical Innovation & Management. Areas of interest include Sustainable Software Engineering, Education Technology, Offline Navigation Systems and Cloud Computing.