IJCSE International Journal of Computer Sciences and Engineering Open Access
Survey Paper Vol.-7, Issue-8, Aug 2019 E-ISSN: 2347-2693

Improvisation of SDLC Model Using Machine Learning Technique (CBR) For Development of Software

Madhup Kumar[1*], Anuradha Sharma[2]

[1]M.Tech by Research, Jharkhand Rai University, Ranchi, India
[2]Department of CSE, Jharkhand Rai University, Ranchi, India

[*]*Corresponding Author: madhup.kumar@bitmesra.ac.in, Tel: +919955346969*

DOI: https://doi.org/10.26438/ijcse/v7i8.241246 | Available online at: www.ijcseonline.org

Accepted: 14/Aug/2019, Published: 31/Aug/2019

***Abstract*—** This paper explores software development through early prediction of planning phase. It summarizes a variety of techniques for software planning prediction in the domain of software engineering. The objective of this research is to apply the various machine learning approaches, such as Case-Based Reasoning and Fuzzy logic, to predict software planning. The system predicts the planning phase activity after accepting the values of certain parameters of the software. This paper advocates the use of case-based reasoning (i.e., CBR) to build a software development prediction system with the help of human experts. The prediction is based on analogy. We have used different similarity measures to find the best method that increases reliability. It can be readily deployed on any configuration without affecting its performance.

***Keywords*—**Software Engineeering,SDLC Model,Machine Learning, CBR

I. INTRODUCTION

Computer have been used for commercial purpose for last 50 years'/w engineering have been useful to solve large and more complex program in cost effective and efficient way with his past experience. We can say s/w engineering is a engineering approach to develop s/w. S/W engineering method based on error prevention, cost effective to prevent error from occurring than to correct them as and when they detected, Well defined stages such as SRS, Designing, Coding, Testing Maintenance ,various design technique used. There are various life cycle models available for developing various types of software. Every SDLC model has some advantages and some limitations. The software developer decides which SDLC model is suitable for his product. The primary advantage of use to a life cycle model is that it .Development of s/w in a systematic and disciplined manner. A life cycle model forms a common understanding of the activities among the s/w engineers and helps develop s/w in systematic and disciplined manner. The objective of this paper is to compare all universally accepted SDLC model and Proposed a new SDLC model for development of software in systematic and disciplined manner.

Previous method that was exploratory method this Based on error correction, error are detected only during the final project testing, in this method there are various limitation like hard to maintain product, break down when used to develop large product. A software life cycle is the sequence of identifiable stages/process that a software product undergoes during its life time. A SDLC model is a descriptive and pictorial representation of s/w life cycle. A life cycle model map the different activities performed on a s/w product from its inspection to retirement .Business organization follow steps-Business process, manufacturing industries-manufacturing process same as for s/w development use s/w process model .The first life cycle of any s/w product is generally feasibility study, RAS, Design, Coding, testing and maintenance. A (software/system) life cycle model is a description of the series of activities carried out in a Software Engineering project, and the relative order of these activities. In this approach there arte several estimation techniques are available for estimating size of the projects i.e. Line of Code ,Function Point Metrics and Feature point metrics same as for project estimation there are different types of metrics are available i.e. Basic COCOMO, Intermediate COCOMO etc. All matrices have some advantages and drawbacks. its depend on nature of the project which metric will be used.

II. OVERVIEW OF MACHINE LEARNING

Machine learning deals with the problem of building computer programs that improve their performance at some task through experience. Machine learning has been utilized in various problem domains. Some typical applications of

machine learning are: Optical character recognition, Face detection ,Spam filtering ,Fraud detection, Medical diagnosis and Weather prediction etc. Major categories of machine learning techniques are: Case-based reasoning(CBR) , Rule induction(RI) , Neural networks(NN) , Genetic algorithms(GA) , Inductive logic and programming(ILP)
Relevant details should be given including experimental design and the technique (s) used along with appropriate statistical methods used clearly along with the year of experimentation (field and laboratory).

A. *CASE-BASED REASONING*
Case-based reasoning is one of the most popular machine learning techniques. Case-based reasoning (CBR) is a problem solving paradigm that is fundamentally different from other major AI approaches. Instead of relying solely on general knowledge of a problem domain it uses specific cases . In place of making association along generalized relationships between problem descriptors and conclusion, CBR is used to predict or estimate for either internal or external attributes of processes, products, or resources. These include software quality, software size, software development cost, software effort, software reliability, software defect and reusability.

Thus, the notion of case-based reasoning does not only denote a particular reasoning method, irrespective of how the cases are acquired, it also denotes a machine learning paradigm that enables sustained learning by updating the case base after a problem has been solved. Learning in CBR occurs as a natural by-product of problem solving. When a problem is successfully solved, the experience is retained in order to solve similar problems in future. When an attempt to solve a problem fails, the reason for the failure is identified and remembered in order to avoid the same mistake in the future. Case-based reasoning prefers learning from experience, since it is usually easier to learn by retaining a concrete problem solving experience than to generalize from it.

Apart from identifying the current problem situation, the central task that all case-based reasoning methods have to deal with is to find a past case similar to the new one. CBR also evaluates the proposed solution, and updates the system by learning from the past experiences. In this process the actual methods, part of the process that is focused, and the type of problems that drives the methods, etc. varies considerably.

- Program Logic

We can broadly categorize the following four primary steps for s/w planning prediction using CBR estimation system:
STEP 1. Retrieve the most similar case or cases, i.e., previously developed projects.
STEP 2. Reuse the information and knowledge represented by the case (s) to solve the estimation problems.
STEP 3.Revise the proposed solution.
STEP 4. Retain the parts of this experience likely to be useful for future problem solving.

Case-based estimation comes in handy when limited domain knowledge is available and the optimum solution is difficult to be defined. In software quality estimation we use analogy by stating, "Similar Projects will have similar costs". An advantage of case- based estimation is that it is easy to comprehend and explain its process to practitioners. In addition, it can model a complex set of relationships between the dependent variables and the independent variables. However, its deployment in software quality estimation needs improvements. The best working example of case-based reasoning is the complex human intelligence. However, our (human) reasoning by analogy is always more than approximate or vague rather than precise and certain.
We present new Case-based reasoning process cycle model to solve our problem(See Figure1).

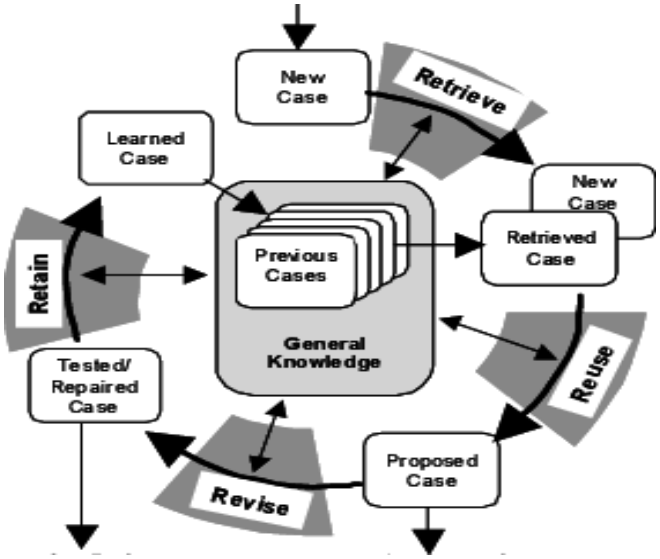

FIGURE.1 CBR PROCESS CYCLE

III. LITERATURE SURVEY

Iqbal et.al., addressed about different development models and their comparison with [1][10]. The paper explained seven different models. The First one is Waterfall model which provides base for other development models. Then its enhanced models are explained in Iterative model, Spiral model, V shaped model and finally, Agile development model. The comparison includes the advantages and disadvantages of different models which can help to select specific model at specific situation depending on customer demand.

Nabil et.al., explain and compare developmental models in software engineering[2]. The paper is concerned with the software management processes that examine the area of software development through the development models, which are known as software development life cycle[11]. It represents five of the development models namely, waterfall, Iteration, V-shaped, spiral and Extreme programming. These models have advantages and disadvantages as well. Therefore, the main objective of this research is to represent different models of software development and make a comparison between them to show the features and defects of each model. Suggesting a model to simulate advantages that are found in different models to software process management. 2. Making a comparison between the suggested model and the previous software processes management models. 3. Applying the suggested model to many projects to ensure of its suitability and documentation to explain its mechanical work.

Bhuvaneswari et.al., illustrated software management processes that examine the area of software development through the development models, which are known as software development life cycle[3]. It represents the development models namely Waterfall model, Iterative model, V-shaped model, Spiral model, Extreme programming, Iterative and Incremental Method, Rapid prototyping model, The Chaos Model, Adaptive Software Development (ASD), The Agile Software Process (ASP), Crystal, Dynamic System Development Method (DSDM), Feature Driven Development (FDD), Rational Unified Process (RUP), SCRUM, Wisdom, The Big Bang Model. These models have advantages and disadvantages as well. Therefore, the main objective of this study is to represent different models of software development and make comparison between them to show the features and defects of each model.

(Kumar et al., proposed The New SDLC-2013 model which is designed in such a way that it allows client and developer to interact freely with each other in order to understand and implement requirements in a better way to produce a high quality software within budget and schedule[4].

As the Software Development process begins with the client's need, so the proposed model tries to discover most of the requirements of the client. It helps in developing an efficient software product that satisfies the client. In the sphere of computer based system products, client satisfaction is dependent on how system development process evolves to build operational product systems that satisfy the perceived and actual client's need and associated system requirements. Ultimately, client satisfaction depends upon the depth of "through-life‟ understanding about the client needs and associated user requirements for a future system, and the ability to communicate these requirements to the system developer. In addition, client satisfaction and confidence depends upon the level of system assurance offered throughout the system development lifecycle. Requirements understanding problems inevitably lead to poor client-developer relationship, unnecessary re-work, and overrun cost and time. The client satisfaction is totally depended on client needs for this reason SDLC-2013 focus on the initial phases.

The proposed work can be summarized as the creation of the approach SDLC-2013 to develop software more efficiently. The aim of Software Engineering is to develop software of high quality within budget and schedule. The proposed plan tries to fulfill the objective of Software Engineering by showing existing matching software as prototype to the client for discovering the requirements efficiently from the client in order to estimate cost, schedule and effort more accurately.

Vishwas et.al., compare and propose a new model developed by incorporating Release Management within the scope of the SDLC basic phases like analysis, design, coding, testing and maintenance[5]. Release Management is the concept which is quite new in the field of Software Engineering. The concept of release management derives itself from the core concept of project management employed in Software Engineering. Software how-so-ever efficient and effective cannot be considered commercially successful until and unless the software remains in the market for sufficiently long duration, in order to recover the cost that incurred during development and deployment of the software. The release management process is a relatively new but rapidly growing discipline within software engineering of managing software releases. As software systems, software development processes, and resources become more distributed, they invariably become more specialized and complex. Furthermore, software products (especially web applications) are typically in an ongoing cycle of development, testing, and release. Add to this an evolution and growing complexity of the platforms on which these systems run, and it becomes clear there are a lot of moving pieces that must fit together seamlessly to guarantee the success and long-term value of a product or project. The need therefore exists for dedicated resources to oversee the integration and flow of development, testing, deployment, and support of these systems. Although project managers have done this in the past, they generally are more concerned with high-level, "grand design" aspects of a project or application, and so often do not have time to oversee some of the more technical or day-to-day aspects. Release managers (aka "RMs") address this need. They must have a general knowledge of every aspect of the software development process, various applicable operating systems and software application or platforms, as well as various business functions and perspectives.

IV. NEW PROPOSED SDLC MODEL

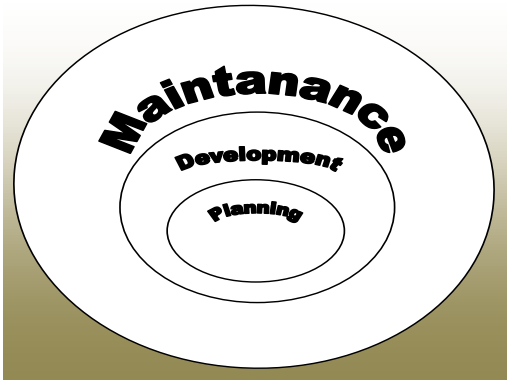

Figure 2. Proposed Model

These are main phases of new proposed SDLC model

I. Planning : As we know how the requirement from customer is important for any software development. There are many software fail or delay due to wrong requirement. The whole process of any software is dependent on requirement gather form user. There are different method form which we can collect requirement from customer or user. The objective of software engineering is that developed software should meet or fulfill the user's requirement. In all prior SDLC model have same problem Estimating beginning of software but this is not in nature how can we estimate accurate or judge size, cost, effort and time of the software before completion of the software . After this we take as this estimated size we calculate Cost, Effort, scheduling etc. this is really become unnatural. These are one of the main causes of project failure or delay.

This is first phase of my proposed SDLC model. In this phase of model CBR technique will be used for project estimation i.e. (Size, Effort, Time and cost). After completion of this phase a document will be produced called project requirement document this is like as SRS document. In this document all required features size of the software, total cost required for development of the software, effort required for the software and time to be develop the software will be measured form CBR technique.

The project manger prepares this document. In this document there are all functional and non functional requirements will be mentioned. After completion of this phase development phase i.e. second phase of my proposed SDLC will start.

For estimating size of the Project I have used Function point Metric. after estimating Size I have used Basic COCOCMO model estimation technique for estimation of Total Effort, Development Time and Total cost of the project.

All the estimation process done in .Net Platform

STEP 1. Retrieve the most similar case or cases, i.e., previously developed projects by single input value i.e. Line of Code.

STEP 2. Reuse the information and knowledge to develop new software.

STEP 3.Revise the proposed solution if any changes required.

And the last step

STEP 4. Retain the parts of this experience likely to be useful for future problem solving.

If not any similar cases are available in DB for any type of the software then I have proposed a new solution .

Step1. Calculating Size of the software by using FPM.
Step2. calculating Effort of the software by using COCOMO Model.
Step3.Calculating Development time of the software by using COCOMO Model..
Setp4. Calculating Total Cost of the software by using COCOMO Model.
Step5. Store all information in DB for future problem solving.

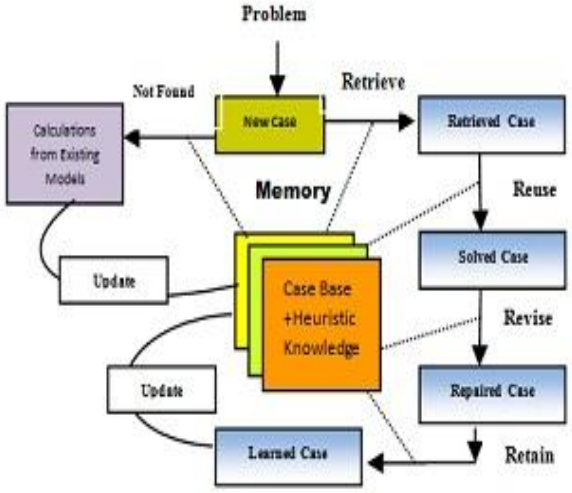

Figure 3.proposed new working model with CBR technique.

II. Development: The main objective of this new developed software life cycle model is to estimate accurate for the software from Planning phase using CBR technique. development phase is second phase of my proposed SDLC. In this phase (a) designing will be done using UML tools to draw UML diagram after that (b) coding will be done using any programming language .

The steps for developing a program are therefore:
a) Understanding the requirement.
b) Produce the design document from PRD document.

c) Translate the design into program code using suitable programming language.

After coding testing will be done. Testing is traditionally used mean testing of program code. When the product is tested with appropriate and realistic tests that reflect typical usage patterns by the intended users, the chances of the product satisfying the customer's requirement is much higher. While testing does not guarantee zero defects, effective testing certainly increases the chances of customer acceptance of the software. Testing is done by a set of people with in a software product (or service) organization whose goal and agreement is to uncover the defects in the product before it reaches the customer. The process of analyzing a software item to detect difference between existing and required condition (i.e., bugs) And to evaluate the feature of the software items.

III. Maintenance: After successful completion of testing phase the last phase of my proposed SDLC maintenance phase will be execute. The term software maintenance denotes any changes made to a software product after it has been delivered to the customer. The maintenance phase of software life cycle is the period in which a software product performs useful task.

Advantages of proposed SDLC Model:

a) Maintainability.
b) Correctness.
c) Reusability.
d) Reliability.
e) Portability.
f) Efficiency.

1. Operation:

a) Correctness: The software which we are making should meet all the specifications stated by the customer.
b) Usability/Learn ability: The amount of efforts or time required to learn how to use the software should be less. This makes the software user-friendly even for IT-illiterate people.
c) Integrity: Just like medicines have side-effects, in the same way a software may have a side-effect i.e. it may affect the working of another application. But a quality software should not have side effects.
d) Reliability: The software product should not have any defects. Not only this, it shouldn't fail while execution.
e) Efficiency: This characteristic relates to the way software uses the available resources. The software should make effective use of the storage space and execute command as per desired timing requirements.
f) Security: With the increase in security threats nowadays, this factor is gaining importance. The software shouldn't have ill effects on data / hardware. Proper measures should be taken to keep data secure from external threats.
g) Safety : The software should not be hazardous to the environment/life.

2. Revision:

a) Maintainability : Maintenance of the software should be easy for any kind of user.

b) Flexibility : Changes in the software should be easy to make.
c) Extensibility : It should be easy to increase the functions performed by it.

d) Scalability : It should be very easy to upgrade it for more work(or for more number of users).
e) Testability : Testing the software should be easy.
f) Modularity : Any software is said to made of units and modules which are independent of each other. These modules are then integrated to make the final software. If the software is divided into separate independent parts that can be modified, tested separately, it has high modularity.

3. Transition :

a) Interoperability : Interoperability is the ability of software to exchange information with other applications and make use of information transparently.

b) Reusability : If we are able to use the software code with some modifications for different purpose then we call software to be reusable.

c)Portability : The ability of software to perform same functions across all environments and platforms, demonstrate its portability

Importance of any of these factors varies from application to application. In systems where human life is at stake, integrity and reliability factors must be given prime importance. In any business related application usability and maintainability are key factors to be considered. Always remember in Software Engineering, quality of software is everything, therefore try to deliver a product which has all these characteristics and qualities.

Table 1.Comparative Study of New Purposed Software Life cycle model with Other Models

Features	***Waterfall***	***Spiral***	***Agile Model***	***New Model***
SRS	Well Understood	Not Well Understood	Not Well Understood	Well Understood
Cost/size Estimation	Low	High	High	Low
Schedule	Within Schedule	Schedule May Exceed	Within Schedule	Within Schedule

Risk Involvement	High	Low	High	Low
User Involvement	Low	Low	Low	High
Guaranty Of Success	Low	High	Good	High
Client Satisfaction	Low	High	High	High
Flexibility	Rigid	Flexible	Flexible	Flexible
Time Frame	Medium	Medium	Medium	Short
Initial Product Feel	No	Yes	No	Yes

V. CONCLUSIONS AND FUTURE WORK

In this research work I compare all traditional SDLC models with each other every model have some advantages and some limitation and I also proposed a new model for software development with different life cycle. In this research work I also compare a new model with other traditional SDLC model. Some of the limitations of different SDLC model can be overcome by this new proposed model but some rest limitations of different SDLC model are not been overcome like late delivery and cost . In my future work I shall improve this new model and try to add more features in this model.

REFERENCES

[1] G. Kadoda, M Cartwright, L Chen, and M. Shepperd. (2000), "Experiences Using Case- Based Reasoning to Predict Software Project Effort", In Proceeding of EASE, p. 23-28, Keele, UK.

[2] I. Myrtveit and E. Stensrud. (1999), "A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models", IEEE transactions on software Engineering, Vol 25, no. 4, pp. 510-525.

[3] K. Ganeasn, T.M. Khoshgoftaar, and E. Allen. (2000), "Case-based Software Quality Prediction", International journal of Software Engineering and Knowledge Engineering, 10 (2), pp. 139-152 .

[4] Shi Zhong,Taghi M.Khoshgoftaar and Naeem Selvia "Unsupervised Learning for Expert-Based Software Quality Estimation".Proceeding of the Eighth IEEE International Symposium on High Assurance Systems Engineering (HASE'04).

[5] Ekbal Rashid, Srikanta Patnaik, Vandana Bhattacherjee "A Survey in the Area of Machine Learning and Its Application for Software Quality Prediction" has been published in ACM SigSoft ISSN 0163-5948, volume37,number5,September2012,http://doi.acm.org/10.1145/2347696.2347709 New York, NY, USA.

[6] M. J. Khan, S. Shamail, M. M Awais, and T. Hussain, " Comparative study of various artificial intelligence techniques to predict software quality" in proceedings of the 10th IEEE multitopic conference, 2006, INMIC 06, PP 173-177, Dec 2006.

[7] S. Becker, L. Grunske, R. Mirandola, and S. Overhage, " Performance prediction of component-based systems a survey from an engineering perspective", In architechture systems with Trust-worthy components, Vol 3938 of LNCS, Springer, 2006.

[8] Ekbal Rashid, Srikanta Patnaik, Vandana Bhattacherjee "Enhancing the accuracy of case-based estimation model through Early Prediction of Error Patterns" proceedings published by the IEEE Computer Society 10662 Los Vaqueros Circle Los Alamitos, CA, in International Symposium on Computational and Business Intelligence (ISCBI 2013), New Delhi, 24~26 Aug 2013 ISBN 978-07695-5066-4/13 IEEE, DOI 10.1109/ISCBI.2013.

[9] Aamodt, A. and E. Plaza, Case-based reasoning: foundational issues, methodical variations and system approaches. AI Communications 7(1), 1994.

[10] Venkata U.B.Challagulla et al "A Unified Framework for Defect data analysis using the MBR technique". Proceeding of the 18th IEEE International Conference on Tools with Artificial Intelligence(ICTAI'06).

[11] Tom M. Mitchell, "Machine LearningSection 4.1.1; page 82, McGraw Hill Companies, Inc. (1997).

[12] David E. Goldberg "Genetic Algorithms in search, Optimization and Machine learning" Pearson Education, Inc.

[13] Luger, George F. Artificial Intelligence, Structures and Strategies for Complex Problem Solving, Fourth Edition, atpage 471 2002. Harlow, England: Addison-Wesley.

[14] Ekbal. Rashid "R4 Model for Case-Based Reasoning and Its Application for Software Fault Prediction," International Journal of Software Science and Computational Intelligence (IJSSCI) 8 (2016): 3, doi:10.4018/IJSSCI.2016070102.

[15] Ekbal Rashid "Improvisation of Case-Based Reasoning and Its Application for Software Fault Prediction" has been published in International Journal of Services Technology and Management (IJSTM).ISSN online: 1741-525X ISSN print: 1460-6720, Vol.21, No.4/5/6, pp.214,227,DOI:http://dx.doi.org/10.1504/IJSTM.2015.073921, Inderscience Publisher.

[16] Catal C. Software mining and fault prediction. WIREs Data Mining Knowl Discov 2012; 2: 420-426.

Authors Profile

Mr. Madhup Kumar is pursuing MTech (by Research) in Computer Science from JRU Ranchi. He has completed his MCA from BIT Mesra Ranchi. He is working as Asssociate Lecturer in Department of CSE, BIT Mesra, Patna Campus.His research area is Machine Learning(CBR),Software Engineering and Networking.

Ms.Anuradha Sharma is an assistant professor in the department of Computer science and Information Technology. She holds a Master degree in Information Technology from Birla Institute of Technology Mesra and a B.Tech in Computer Science from B.P.U.T (Biju Pathnayak University of Technology). Her topic for research while M.Tech was "Prediction of Iron ore using Soft Computing Technique" which involved Artificial Neural Network using Back propagation method. Her area of interest include Software engineering, RDBMS, soft Computing, Neural network. She is an avid reader and holds interest in current affairs topic.

