Proposal of Software Development Model Using Fuzzy Logic

[1]Rishi Raj, [2]Rajat Mittal, [3]Subhash Chand Gupta, [4]Tanupriya Choudhury
[1,2,3] Amity University Uttar Pradesh, Noida, India
[4]Assistant Professor, University of Petroleum & Energy Studies (UPES), Dehradun, India
[1]Raj231298@gmail.com, [2]rajatmittal100@gmail.com, [3]scgupta@amity.edu, [4]tanupriya1986@gmail.com

Abstract— the defects in software development life cycle model has recently attracted the instinct of many researchers. It is very important to identify the defects or various risks in every phase of life cycle of software development. It is useless to detect the defects in testing phase of SDLC model as it cost heavy effort and manpower to get resolved. Therefore the proposed SDLC model is of great use. The main objective behind this of this paper is to propose a software development life cycle model using the fuzzy logic which will eliminate the risk of software failure with the introduction of new selection phase in the software development life cycle model. A fuzzy logic based team selector has been introduced between the requirement and designing phase based on experience of the teams. As result the team selector phase will reduce the failure which will be identified in later phases of SDLC model. The fuzzy logic based selection of team in the team selector phase will reduce the risk of failure of the software and hence the defects or risk identified in testing phase will decrease as the team which is best for the project will be employed.

Keywords-SDLC model; defects; fuzzy logic; team formation

I. INTRODUCTION

This in the recent days, human dependence over the software has directly or indirectly been influenced. The consequences of software failure has been seen from the day when first software was ever made. Software development is referred as the continuous evaluation of the requirements of software on the basis of current status of software. If the software projects continue to fail in this order then this will result in the losses of humans and money. Therefore the need of a better software development life cycle model is essential in software development process.

Generally, the software projects are shut down even before they are ever started. According to a study, only 3 project out 100 are successful and a large amount software are restarted due to lack of requirements and sometimes project reach at a stage where there is no way to move forward or go back. Software development are done my human so they are measured by man-hour. We all know that the man-hour is of fuzzy nature. To ensure the failure free software is very challenging. Therefore there is an increasing need to develop a phase which can determine a project specific team. Few evolutionary models has been proposed in the software development which has its own advantages and disadvantages as well.

Software development life cycle model consist of various evolutionary models in which common models are prototype, spiral and waterfall models, each having their own benefits. Waterfall model consists of requirement analysis phase which gathers the requirement needed for the software project and provides the software requirement specification (SRS) as a input to the designing phase.in the same order the designers use the SRS to design the software and provide the software design description(SDD)to the next phase of life cycle model. Once the software is designed then the programming is done of the module and unit testing is performed. After the integration of different module is done then overall system testing is done in order to check the functional and non-functional working of software. Last phase of the life cycle model is the maintenance phase which include the addition of extra feature or removal of faults and errors. Similarly the prototype model performs the same task as by waterfall model but with a introduction of quick design phase between requirement and designing phase which involves the customer in the software development process. Once the rough design of software is approved by the customer then only the designing phase is initiated. Spiral model is the continuous evaluation of project with graduation of same phase again and again but with additional risk analysis phase.

The above mentioned software development life cycle model has their disadvantages i.e., lifecycle model has disadvantage that it is very difficult to get all the requirement at the initial phase of the software development. Prototype model has a disadvantage of requiring a knowledgeable customer which cannot be taken for guarantee. Similarly spiral model requires experienced employee in a way to determine the risks involve in process for the development of the software and finding out the constraints. Requirement of experienced employee can cost too much for an organization for a low budget software project. Therefore in this paper, software development life cycle model using fuzzy logic is proposed in order to overcome these problems.

II. LITERATURE REVIEW

The main aim for which SDLC model is used is to enable all the efforts that can be utilized to make all phases of project run efficiently.

R Akalya and K. Meera [1] (2014) software project's productivity and development depends upon numerous co related factors.

978-1-5386-5657-0/18/$31.00 ©2018 IEEE

IEEE [2] (2004) specific requirements and challenges of a project can evaluated using SLCM selection and after that best model can be chosen.

T Choudhury [3] (2016) to make software from defect free, software testing is used as a process to find errors and remove them.

L. Zadeh (1965) scientific term that is used for mathematics developed model for unique way of processing words by brain is Fuzzy logic [4]. It is explained as ‘‘As complexity rises, meaning of precise statements are lost as the complexity rises and meaning and meaningful statements loose precision”.

E. Mamdani (1974) FLS is most common and widely used with fuzzifier and defuzzifier where the work of fuzzifier is to map crisp inputs into fuzzy sets and the work of defuzzifier is to only map fuzzy sets into crisp outputs where ever required [5]. Fig 1 shows the logic system which is proposed by Mamdani.

ISO/IEC 9126 [6], Geraci A. and Katki F [7], Abowd G., Beale R. [8], Donyaee M. and Seffah A [9], McCall J. A. and Richards P. K [10], T Choudhury [11], [12] extends about different standards and we examined a huge number of international standards and different usability models, in which description of various properties in non-homogeneous manner is given. Thus creates lots of confusion between experts who studies there applications. Few researchers have also tried to use the fuzzy logic in order to propose the defect density in all the phases of life cycle for software development.

III. PROPOSED MODEL

In the introduced SDLC model the fuzzy logic based team selector phase is introduced. It is very necessary to determine or allot the correct team for a particular software project based on the requirements. This model will change the process in which the earlier model used to develop a software. The proposed model contains the following phases in the software development process:-

1. Formation of team
2. Requirement Analysis
3. Selection of team
4. Evaluation phase

The architecture of the proposed software life cycle model is figured in Fig.1.

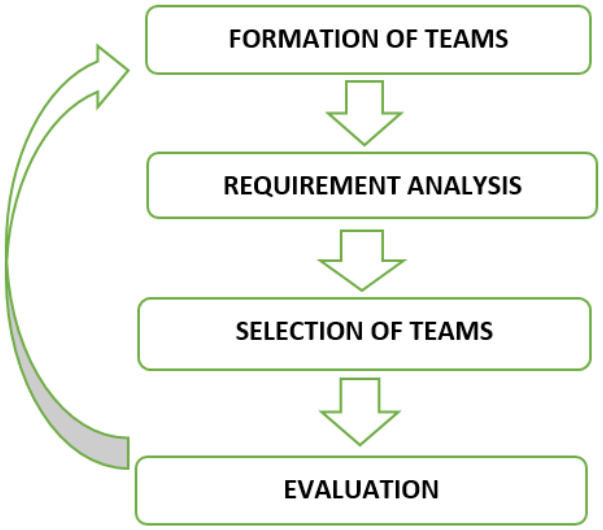

Figure 1. Proposed model

1. Formation of teams

Team formation is a vital phase in any software development organization. In this proposal, the basis for team formation is done with the experience of the team members. Each team will consist of equal number of software requirement analyst, designers, programmers, tester and a manager who will work as a bridge between all the members of different domain and customer. Each team will consist of a well assembled team members with a combination of low experienced, average experienced and highly experienced members. One of the team formed in proposal is given in table 1.1, 1.2, 1.3.

TEAM A	LOW EXP	AVERAGE EXP.	HIGHLY EXP
SOFTWARE ANALYST	1	0	0
DESIGNER	1	0	0
CODER	1	0	0
TESTER	1	0	0
MANAGER	1	0	0

Table 1.1 team a categorised into low experience set of members

TEAM B	LOW EXP	AVERAGE EXP.	HIGHLY EXP

SOFTWARE ANALYST	0	1	0
DESIGNER	0	1	0
CODER	0	1	0
TESTER	0	1	0
MANAGER	0	1	0

Table 1.2 team b categorised into average experience set of members

TEAM C	LOW EXP	AVE AGE EXP.	HIGHLY EXP
SOFTWARE ANALYST	0	0	1
DESIGNER	0	0	1
CODER	0	0	1
TESTER	0	0	1
MANAGER	0	0	1

Table 1.3 team c categorised into highly experience set of members

In the similar fashion, number of team will be formed with the various combination of experience of the analyst, designer, coder, tester and manager.

2. Requirement analysis

Analyzing the requirements at the beginning of the project is not an easy task. A software should contain its required functions. This is the phase in which we get the overall requirements of the software in order to divide them in the proposed category [3] (given below). There will number of experienced employer who will be employed in order to divide the requirements in proposed category which will act as an input towards the fuzzy output.

Proposed categories:

SR: Similar with the existing software requirements

DR: Different with the existing software requirements

SD: Similar design like existing one

DD: Totally different design from existing software

Suppose if a project arrives with maximum similarity with the existing developed software then it will be put into DR, if any project arrives with no similarity then it will be put into DR, if in any case the designs of s project are similar from any existing software and does not requires special emphasis then it will be put into SD, and if the designs required are totally different from the existing one and requires a special emphasis then it will put into DD category. These categories will act as an input to the next phase in order to determine that which team will work on a particular software development project [14].

3. Selection of team

After the requirements are analyzed we have our motive. We have categorized the requirements in the proposed categories in order to provide the inputs to the selection of team phase. In this phase we will select a team which is formed in team formation phase based on the necessity of requirements gathered in requirement analysis phase using the fuzzy logic [13]. The rule which will be employed in the selection of team will work on simple if-then-else cases i.e., we will be selecting a value for fuzzy range between the requirements categories. The formula proposed in this phase is defined below:

w(t)=max[SR, DR] and
z(t)=max[SD, DD]

Rule 1:
w(t)=SR and z(t)=SD
Then Team A will be selected
Rule 2:
w(t)=SR and z(t)=DD
Then Team B will be selected
Rule 3:
w(t)=DR and z(t)=DD
Then Team C will be selected

Fuzzy algorithm for selection of team on the basis of requirements-

As extreme cases of truth, 0 and 1 is included by fuzzy logic. However additionally includes the assorted states of truth in between so, as an example, the results of a comparison between any 2 things such as for property of tallness, might be not "tall" or "short" however ".54 of the property for tallness."

Working of the fuzzy selection algorithm is shown in the figure 2.

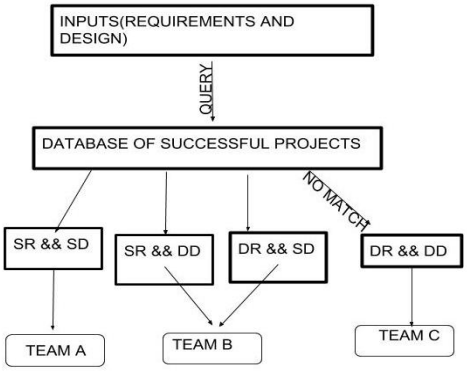

Figure 2.working of fuzzy selection algorithm

```
PROCESS OF SELECTION OF TEAM
while(requirements!=0)
{        inputs(requirement, design);
         check();
          selection(req, des);
}
check(){
          if(requirements=same && design==same)
           return SR,SD;
          else  if((requirements==same  &&  design==different)
||(requirements==different && design==same))
          return SR,DD;
          else
                    return DR,DD;
}

selection(req, des) {
                              if(SR && SD)
                         select team A;
else if ((SR  && DD) ||  (DR && SD))
select team B;
else select team C;}
MEMBERSHIP VALUES
{SR,SD,DR,DD}
```

Membership graph of different teams are shown in figure 3, 4 and 5 respectively.

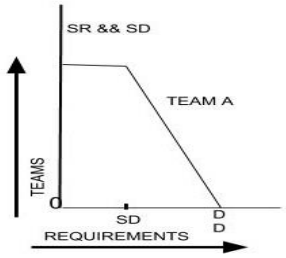

Figure 3.membership graph of team A

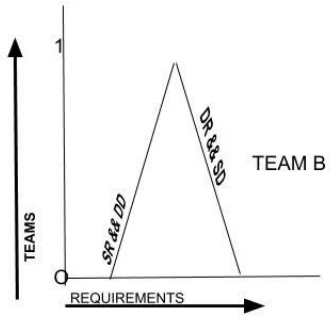

Figure 4.membership graph of team B

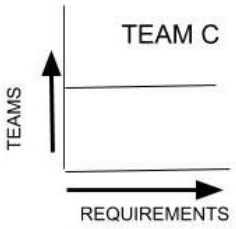

Figure 5.membership graph of team C

Based on the above formula and rule we can determine the team necessary for the proposed project for software development.

4. Evaluation

After the successful allotment of teams toward their project, there is a need for continuous evaluation of the work by the customers in order preserve changes or maintainability of the software which is developed by the team along with the managers. Continuous evaluation of the project can eliminate the risk of failure of the project. In this way the customers as well as higher employees can keep track of job done or being processed.

Advantages

- Only the required amount of effort and time will be employed with help this model.
- For a similar existing low budget project can be handled by low experienced team which will minimize the cost.
- Only highly experienced employees will employed to the best project in order to maximize the chances of success.

IV. CONCLUSION

A software development life cycle model has been proposed using fuzzy logic. It's not easy to deploy a software life cycle model in recent days. The proposed SDLC model redefines the working of the evolutionary SDLC models such as spiral,

prototype and also waterfall models. In the earlier model we do not have a team based on the requirements and design architecture, which was fulfilled by this fuzzy based model. In this SDLC model we have used the concept of fuzzy to select the appropriate team for appropriate project, this will not only reduce the failure of software but will also reduce the cost which was previously wasted by application of experienced or any employee to any project. In this SDLC model we have used fuzzy logic with few rules to determine the teams which was formed in accordance of the first phase of the model. In this paper the proposed SDLC model is applicable on all types of project, either small scale or large scale.

V. REFERENCES

[1] K. Meera, "A Comparative Study of Software Effort Estimation Using Fuzzy Logic Membership Function," International Journal of Innovative Research in CCE, vol. 2, no. 1, March 2014

[2] Kahraman C. and Ruan D., 2004: A fuzzy multi-criteria decision approach for software development strategy selection, International Journal of General Systems, Vol. 33 (2–3), pp. 259–280.

[3] T Choudhury, 2016: "A walk through of software testing techniques". SMART international conference, IEEE, 2016.

[4] L. Zadeh, "Fuzzy sets, Information and Control," 1965

[5] E. Mamdani, "Applications of fuzzy algorithms for simple dynamic plant," IEEE, 1974.

[6] ISO/IEC 9126: Information Technology-Software Product Evaluation-Quality Characteristics and Guidelines. 1991

[7] Geraci A. and Katki F., IEEE standard glossary of software engineering terminology.

[8] Abowd G., Beale R., Human-Computer Interaction, 2nd ed. Prentice-Hall, 1998

[9] Donyaee M. and Seffah A: An integrated model for specifying and measuring quality in us, Eighth IFIP Conference on Human Computer Interaction, Tokyo, Japan. 2001.

[10] McCall J. A. and Richards P. K, Factors in software quality, Rome Aid Defense Centre, Italy, 1977

[11] T Choudhury, "Intelligent Classification of Lung & Oral Cancer through diverse data mining algorithms"

[12] T Choudhury, "A Smart Approach to Diagnose Heart Disease through Machine Learning and Spring leaf Marketing Response