International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 6, Issue 6, November- December 2017 ISSN 2278-6856

A COMPARATIVE ANALYSIS OF SDLC MODEL WITH SECURE SOFTWARE DEVELOPMENT AND TESTING

Nandita Tiwari[1], Dr. Varsha Namdeo[2]

[1]Computer Science and Engineering Department, SRK University Bhopal, India

[2]Computer Science and Engineering Department, SRK University Bhopal, India

Abstract

The complicated and the emergent nature of the software systems make the environment of the data and its key components very unpredictable. The increasing demand of the software with cheaper cost, quick delivery and high quality than the former, has made the software development more complex and diverse. The software development models are useful for developing the software in a systematic manner and delivered within the constraint deadline. In this paper, starting from the initial liner model to the latest used model for software models is described. Comparative analysis of the models is done to show which is better against which parameter. The secure software development approach is also discussed which combines the different tools and approaches to make the software code less vulnerable to attackers. Latest used methodology agile methodology is described which make software development and user interaction more easy. After the development of software, testing is done with the various test cases to make it error free. Different types of the testing techniques such as Chi-squared test, Student's t-test, G-test, Unit testing and Wald test with the implementation formula are described.

Keywords: Software development life cycle (SDLC), Waterfall model (linear sequential), Incremental model, Prototype model, Spiral model

1. INTRODUCTION

Software engineering is a structured, methodological and coherent approach used for the growth, performance and maintenance of the software systems. It main aim is to address the poor quality of the software. When different people put same type of methods on software, alike methods will be created. When this mechanism is applied to software systems they easily meet the constraints of time, quality and cost. SDLC in the software engineering is described as the process of building systems, models and techniques which are used for their development of products. SDLC offers the linear operations for the software developers to create the product in a manner that it should of high quality and completed within the time constraint.

Software engineering is the branch related with growth of software product by using accurately defined scientific principles, approaches and techniques depicted in Fig. 1. The result of the software engineering is an effective and consistent software product. Software project management has broader possibility than software engineering process as it includes communication, pre and post-delivery support etc. This software engineering will offer the user understanding of software product, software design and development process, software project management and design complexities etc.

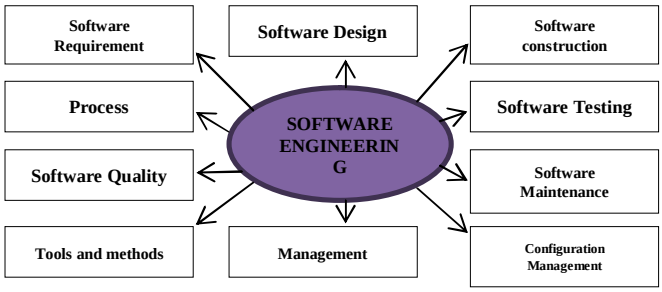

Figure 1 Software engineering

The term software engineering is the addition of two terms software and engineering. Software is the combination of many executable programs, documentation and associated libraries. When the software is made up for a specific need it is known as software product [1]. Whereas engineering is about the development of products, using correctly and structurally defined principles and approaches. The German computer scientist, describes software engineering as the establishment and use of sound engineering to obtain economically efficient software which work well on the actual machines.

2. SDLC (SOFTWARE DEVELOPMENT LIFE CYCLE)

The main purpose of the design review is to prioritize the design task so as to make sure the users understand the design and users can check that the development team recognize the needs easily and efficiently. In this way, it is conformed that the project is on right track.Software development life cycle is a cyclic methodology, the stages also known as phases repeat; hence alterations can be made to design in the following cycle. So this design is less rigid as compare to other linear methods. This is the main reason why the methodology should be known upfront as the approach for every phase get changed according to methodology. A lot of time should be given to the design phase as the whole project planning and process is dependent on this phase. After the collection of all the

requirements, the design process starts.The users should be incorporated in the phase by taking feedback to understand the needs of the users. Before making all the collaborative decisions of design phase, firstly all the needs should be understood and met. The project success should also be known [2].
Organizations continually adapt there IS (information systems) to reflect alterations in the type of information needed. There alterations are due to change in technology, the organization's structure, the organization's business processes or any other external environment.

The systems development life cycle has been designedto guarantee that the alterations are orderlyproductive.
There are many activities in the process of software developmentwhich can be distributed into sections called phases. These phases can be evolved into the software development as per requirement. Mainly there are 5 steps in software development life cycle described in table 1:

a. Requirement Gathering and analysis
b. Designing
c. Coding
d. Testing
e. Maintenance and Support

As depicted in Fig. 2, the initial stage comes up with many responsibilities: like gathering of all the requirements from the users and analysing them if they are feasible to get developed. Second phase is designing in which the whole plan to solve a problem is decided. Next is the code phase, here the planned solution is implemented on the hardware using the software coding modules. For checking the software reliability, consistency and error free nature, test cases are applied on the software product in the testing phase. At final, deployment of software is done on the hardware of the user and maintenance steps are further carried out by user according to their requirement [3].Each phase in the software life cycle has its' own tasks and the deliverables which are fed to next phase.

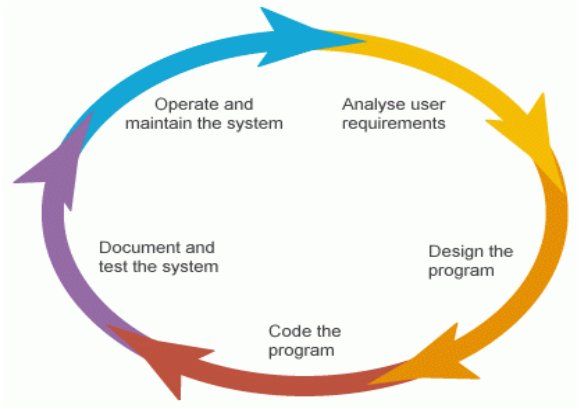

Figure 2 SLDC phase

Table 1 Phases of Software development life cycle

S.No	Phases	Description
1	Requirement gathering	• This is the main critical task for the success of project. • Every requirement need to be fleshed out in detail. • Iterative process is defined as the one in which lot of interaction takes place between the stakeholders, users and the project team by conducting interviews and different surveys. • Many use cases are described for each action of the new system.
2	Design	• Technical design needs are prepared by the architects and lead developers. • Business requirements clarify how the application be written. • Technical requirements clarify which tables to be added, any new transactions which need to be added etc. • Specification of every small thing like the definition of data entry field, time-zone and work-flow of every button to be clicked. • Extensibility of the product is declared • Resource and data constraints is defined
3	Coding	• Actual coding and unit testing is done. • If an alterations are introduced the developers need to be open minded and supple.
4	Testing	• Testing is performed through the test cases. • Different types of testing are performed like integration and system testing. • User acceptance testing is the final stage of testing and is performed by end users to ensure that product is according to their requirement. • Once all the testing is done, implementation and deployment starts.
5	Implementation/ Deployment	• Deployment complexity is dependent on the project size. • Training is given to the users and operations staff

There are many models in the Software development life cycle. The water fall is the initial model which is very traditional. In the waterfall model a well ordered plan and requirements need to be followed. This method works well for the large projects which have time to develop. Other one is agile method which is flexible in requirements, design and coding process and is iterative. Agile method works vest for the smaller projects and is very adaptable to continuous improvement in the application according to users' need [4].

2.1 Waterfall model

It is a linear sequential model also known as waterfall model. This is the oldest model in the software engineering and is used in many government projects. All the planning is done in the initial stages so the design flaws can be known beforehand. If the quality control is a major concern in the project, then waterfall model is best suited. A pure linear waterfall model involves many non-overlapping stages [4]. The merits and demits with usage is describe in table 2.

Table 2: Description of waterfall model

Description	Advantages	Disadvantages	When Used	Applications
Linear path is followed	Easy to implement as it is sequential	Problems in any of the phase ae not completely solved at that phase which leads to many other problems.	Requirements are well known and clear	Commercial projects
All features are previously planned for simultaneously implementation	Resources needed to implement this model is very small	After the process of development has been started and the client want to change the requirement, the implementation will not be done again	Stable product definition	E-commerce websites
Design and implement all features	Proper need of requirement and documentation is required for the high quality development	(Freezes scope) Customer requirements contract need to be free-zed before the development starts	Technology is well understood	Portal
Test all features	Cost and the resources are beforehand known	The delivery time of the project is also estimated and scheduled in advance.	No ambiguous requirements	Network
No overlapping is done in the phases and every action or job need to be completed in the specified period.	Every task is planned previously	No go back links to previous stages.	Small scale projects	

2.2 Incremental model

This model combines the components of linear model with the iterative prototyping. The linear sequenced phases are applied in staggered structure as the time progresses. Every linear sequence creates a deliverable increment of the product.The prototype is repeatedly developed depending on the changing requirements of the users until they get a desired software product. Requirements are confined into many modules and every module is independently designed and developed [5].Every module of the project is designed using the waterfall model.

For example, a word processor software product is produced with its first increment having the basic facilities like editing, management and other document functions; more improvised editing and management capabilities in the second increment; grammar and spelling checkingin the third increment. When this type of model is used the initial development is the core product. The merits and demits with model usage is describe in Table 3.

2.3 Prototype model

The main idea of the prototype model is that a spontaneous prototype is created to fully understand the requirements of the user. The system prototype is based on current user requirements.By the use of prototype, user gets the feel of actual system. The interaction between the client and developer is common so as to understand the users' requirement seriously. Prototype model is the better way for knowing the requirements of the large systems [4, 6]. The merits and demits with its usage is describe in Table 4.

Table 2: Description of prototype model

Description	Advantages	Disadvantages	When used
Prototype is created to fully understand the requirements	Users' involvement in every task	Only forefront is implemented first	When system needs a lot of interaction

of the user			
Not a complete system but as many of the systems are not developed	Working system is provided to users to get clear idea	Less functionality at starting	In online systems and web interfaces
-	Delusion perceived much before	If one module does not get completed, after full designing it may get complicated to evaluate and complete it	System need to get changed according to the users requirements
-	Feedback is rapid which lead to better solutions	Complexity increases if the scope of system increases beyond expectations	-
-	Functionality related problems can be recognized easily	Incomplete problem analysis	-

2.4 Spiral models

This model was originally created by Bohem. It is an evolutionary process model that couples the repetitive form of prototyping with the linear nature (systematic and controlled) of the water fall model. It gives the potential for the fast development of the incremental versions of the product.The software is released in incremental issues. At initial iterations, the further releases may be as a paper model.After the initial iterations, more versions of the system are produced.

This model is divided into many activities which are called as task regions. The task regions can range between 3 and 6. The task regions of spiral model are described in Table 5 and its whole description with merit and demerits are described in Table 6.

Table 5: Task regions in spiral model

Task regions	Description
Customer communication	Tasks are created to create an effective interaction between the developer and the user.
Planning	Tasks are established to describe resources, schedules, timelines and other product related information.
Risk Analysis	Tasks for both technical and management issues
Engineering	Tasks needed for building representation of application which can be one or more.
Construction and release	Tasks needed to build, test, install and give user support.

Table 3: Description of spiral model

Advantages	Disadvantages	Usage
As thee is high amount of risk analysis, high risks and issues are avoided	Model is costly when used with many iterations	In large and mission critical projects.
Better for high criticality projects	For the risk analysis, highly specific expertise	When cost and evaluation of risk is very important
Strong approval and documentation control	Project success is fully dependent on the risk analysis phase	When long term commitment is not so important as the potential changes occurs in the cost and resource priorities
Software is created at the early stages	Model is not good for smaller projects	When the customers are uncertain of their needs
Extra functionality can be added later	Risk evaluation should be previously done	Complex requirements and new product line is launched

3.COMPARATIVE ANALYSIS OF THE DIFFERENT SDLC MODELS

Depending upon the various parameters (requirement specification, cost, risk analysis, time, flexibility), the software development life cycle models are compared in Table 7 [7].

Table 7: Comparison between the SDLC models based on the parameters

Traits	Waterfall model (linear sequential)	Incremental model	Prototype model	Spiral model
Requirement specification	Done at the beginning stage	Done at the beginning stage	Changes frequently	Done at the beginning stage
Understanding requirement	Fully understood	Not understood well	Understood well	Understood well
Availability of reusable component	No	Yes	Yes	Yes
Complexity of system	Modest	Modest	Multifaceted	Multifaceted
Risk analysis	At initial stage	No analysis of risk	No analysis of risk	Yes
User involved in all phases of life cycle	Only at initial part	Intermediate	High involvement	High involvement
Guarantee of success	Less	High	Good	High
Overlapping phases	No overlapping in the phases	No overlapping in the phases	Yes	Yes
Implementation time	Long	Less	Less	Depend on the project
Flexibility	Stiff	Flexibility is less	Highly supple	Flexible
Changes incorporated	Tough	Easy	Easy	Easy
Expertise required	High	High	Medium	High
Cost control	Yes	No	No	Yes
Resource control	Yes	Yes	No	Yes
Cost [8]	Low	Low	High	More costly

4. SECURE SOFTWARE DEVELOPMENT

The secure software development is a methodology in which software related improvement designs ideas are connected.

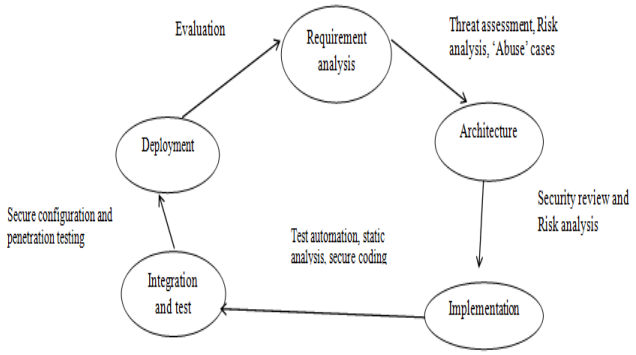

Figure 2 Security tasks in between different Phases in SDLC

It contains all the enquiries about the programming product and also the requirement about the software which then helps the product to assemble [9].It comprises of: (a) requirement gathering (b) software design (c) programming. Software assurance is needed in numerous sectors like defence department. Many of the vulnerabilities occur in the application layer. Organization should take bold steps to ensure the integrity, availability and confidentiality of the software code. It was a myth previously that network infrastructure can guard against the malicious attacks [10]. The intruders are very much successful in cracking the code to enter the channels through the cross site scripting, buffer overflow exploitation and many more. Different test and risk analysis are done on different phases to protect the code from the attacks and make it more secure which are described above in Fig. 3.

4.1Agile Methodology

The Agile approach is cyclical in nature and is based on iterative and incremental development, where requirements and solutions evolve through collaboration. Due to the cyclical nature, it allows for problems to be resolved as phases repeat. It allows for issues to be found and then addressed in the next cycle [11].Software Development Life Cycle (SDLC) consists of few phases like planning, analysis, design and implementation. Many number of SDLC models have been developed like Waterfall, Spiral and SCUM etc. There arenumerous number of new methods, SCRUM (Agile methodology) is one of them. Agile consists of many methodologies but SCRUM is most famous and powerful methodology which provides benefit to companies. SCRUM is simple for managing difficult projects. SCRUM is an agile process. It is used at many companies with success when compare to traditional SDLC

model [12]. The main chief principles of agile methodology are discussed in the Table 8.

Table 4: Chief principles of agile methodology

S.no	Key principals in agile method
1	Face to face interactions are important
2	With the scalable model, software can be adaptable to changing needs
3	Accurately working product is a measure of success
4	Simplicity is important
5	Regularly delivered software
6	Unavoidable user evolvement
7	They software team and the software needs are better reflected as the software grows

5. TESTING TECHNIQUES

After the software development, testing is the vital phase which ensures the accurate working of the software. Every condition is individually tested first. After that, the condition which make up one module, dependent conditions, follow up conditions, two linking conditions are tested one by one with the previously created test cases based on the problems or errors that generally occurs in the software. All the test cases, for the condition are previously created based on the type of conditions and constraints applied on the software product. Some of the test techniques are describes as follows:

Chi-squared

It used the sampling method to recognize the project data warehouse a project plan standard which are nearly linked to existing analysis.

When the project plan is selected by the hybrid chi-squared method, the monitoring tasks begin to check the accuracy of the project plan choice. The main aim after creating this method is to offer a consistent analysis of data to identify superior configurations, intended to support decision task and detecting the deviation problem in the project plan activities.

The method created and adapted based on Chi-Squared integrated qualitative and quantitative info, by usage of calculations of descriptive statistics. This method was applied exactly to govern and monitor the inconsistency of deviations found in projects in development environment.

When Chi-Square test was performed via evaluations, two variables were used which are: OF= Observed Frequencies, from a set of software testing, to recognize categorical variables, and the EF = expected frequencies, thusfinding the primary indicators.

The designed algorithm assumed that the variables havedirect relationship between them. The variables selected for test followed the simpledistribution. The formula for calculation of chi-square is described in Fig.1.

$$X^2_{calc} = \sum_{i=1}^{r}\sum_{j=1}^{s} \frac{(OF_{ij} - EF_{ij})^2}{EF_{ij}} \qquad [1]$$

G-test

G-test are increasing used in the space of chi-squared test. These are the likelihood ratio or maximum likelihood statistical significance tests.

The general formula for the G-test is described In Fig.2.

$$G = 2\sum_i O_i \cdot \ln\left(\frac{O_i}{E_i}\right) \qquad [2]$$

In the equation Fig.2, there are some terms which are defined as follows:

O_i = Detected count in cell

E_i = Predicted count under the null hypothesis

$\ln$ = Sum is taken over all non – empty cells

Students' t-test

Students t-tests' is a statistical hypothesis test that contains test statistics which follows a student's t-distribution under the terms that there is no resemble between the sets.

This test is mainly functional when the test statistics use the normal distribution if the rate of the scaling term were known beforehand. It can be used where we have to find whether the two sets are different from one another. The formula of the student's t-test is described in Fig. 3.

$$t = \frac{z}{s} = \frac{(\bar{X} - \mu)/(\sigma/\sqrt{n})}{s} \qquad [3]$$

In the above Fig. 3,

$\bar{X}$ = Sample meanfrom $(X_1, X_2, X_3 \ldots . X_n)$,

Where n= size

$$s = \frac{\text{Sanmple standard deviation}}{\text{Population standard deviation}}$$

σ = Population standard deviation of data

μ = Population mean

Unit testing

It is the form of software testing by which the every set of software is tested individually as well as tested when combined with the associated modules to find which is fit for use.

It is the level of software testing where the components of the software are tested. The main purpose is to check that every unit of product is developed properly. The smallest modules called as unitis the smallest testable part of the product.

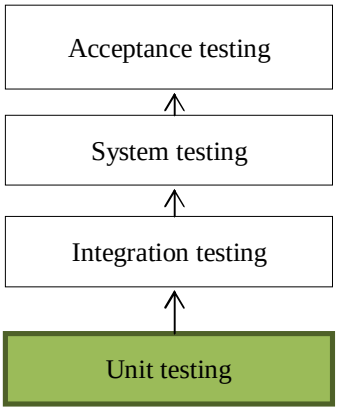

Figure 3: Unit testing

There are few inputs but single output. In the technical programming, a unit can be a function, procedure or function. In the OOP (object-oriented programming), the method is the tinniest unit which belong to super class, abstract class or child class. It is the initial level of testing and is performed before the integration testing.

It is mainly executed by the software developers or in many rare cases; it is performed by the software testers.

There is a schedule for the test plan/ test case in the unit test plan:

- Prepare
- Review
- Rework
- Baseline

At last, the unit testing is performed. There are many benefits of the unit testing which are follows.

a. Change and maintenance in code becomes easy.
b. Codebecomes easy to re-use as the developers knows which modules are independent.
c. Development becomes faster.
d. Cost of fixation of defect in the unit testing is less as compare to higher levels.
e. Reliable code
f. Easy debugging

a. Wald test

Wald test is a kind of parametric statistics test that assumes that trial data comes from a population that goes with the nature of probability distribution which is based on a static set of parameters.

In the Wald statistical test, the universal case formula is described in Fig. 4. In this set, the maximum likelihood estimate of the interest parameter is compared with the given value. The assumption is taken that the difference between the 2 will be normally distributed.

$$\frac{(\check{\theta}-\theta_0)^2}{var(\check{\theta})} \qquad [4]$$

$$\check{\theta} = Maximum\ likelihhid\ estimate$$
$$\theta = Parameter\ of\ interest$$
$$\theta_0 = Proposed\ value$$

6.RELATED WORK

A lot of research has been done in the area of software engineering for the development of projects (major/minor). Few of the researches are listed below defining the effect of models on software engineering by means of coding, reports, approaches of software development and so on.

Helen Sharp et al. [1],have explained how the ethnography benefitted the empirical software engineering researchers. This process was achieved by elucidating 4 characters that ethnography can perform in advancing the aims of empirical software engineering: to reinforce investigations into social and human traits of software engineering; to notify the scheme of software engineering tools; to progress method development; and to notify research programmes. Ethnography usefulness to software engineering has been discussed**. Valerio Cosentino et al. [2],**have surveyed on the data mined from GitHub. The software engineering different aspects has been reviewed. The main merit of the survey was that, it identified the amount, topic and various empirical methods of the study which aimed at how software development practices were influenced. More focus was on the interaction about coding associated tasks and the project communities. Different issues related to project management and community aspects of software development has been described.**Joanna f. Defranco and Philip a. Laplante [3],** have surveyed the type and superiority of research executed on software development team communication. 184 research papers have been analyzed and software engineering team research taxonomy has been created. This taxonomy has been utilized to classify the context of research papers. The outcomes showed that most lively/energetic software development team communication research areas were global software development, effective teamwork and project effectiveness.**MojtabaShahin et al. [4],**have reviewed different approaches related software development. Different tools used for developing software and challenges which the user and the programmer come up with have been reviewed. The systematic literature review technique has been used for peer-reviewing papers. For analysing 69 papers, the data thematic analysis technique has been used.**Iain Bate et al. [5],** has developed new bailout protocol which guarantee the high criticality software process. Integration of model with the current existing techniques has been showed. The merit of the proposed algorithm was that it allowed flooded (over run) of high criticality jobs to be taken by the non-execution of LO-criticality jobs. The amounts of non-executions were though kept minimum. Furthermore, high criticality jobs never fails, loans were taken by HI-jobs and repaid by low criticality jobs. The outcomes showed that proposed bailout protocol is extremely effective in minimizing the quantity of time spent in the HIGH-criticality mode and consequently in minimizing the number of LOW-criticality tasks abandoned to guarantee precise HI-criticality behaviour. **José Luis Fernández-Alemán et al. [6],**have presented the outcomes of two educational experiments which were carried out to find if the process of requiring requirements has an influence on efficiency and

productivity in distributed and co-located software development environments. The main aspect of the experiment was the evaluation of RE learning method based on catalog and RE learning method based on specific requirements. Global results showed that an development can be made to the specification process by using learning based on Reusable Requirements Catalogs (RRC)such as I-CAT.

2. CONCLUSION

This paper contains the description of different software development life cycles used in the development of the software product. At initial, waterfall models were used in which all the planning and requirement gathering need to be collected at the beginning which is not feasible with the current software product needs. Hence, many needed advancement were done in the software models to make it feasible with different projects and user requirement. Spiral model was mostly used in the project development which takes more user interaction and gives the software product in its first increment and with changing requirement and active user interaction, product becomes more accurate with the period of time. But with this, software development vulnerabilities also occur, so secure software development (SSD) model must be followed to make the software planning, code and advancements more secure and attackers are not able to find what developer is up to or after installation what user is doing. At present, agile methodology is used in most of the project as the user interaction, feedback, adaptability, security and advancements are more in this method. After development of test cases testing is carried out, test cases are created for every small condition. Mathematically or statistically based test techniques are applied like chi-squared test, G-test, Ward test. Every test has some different formations based on different conditions.

References

[1] Sharp, H., Dittrich, Y., & de Souza, C. R, "The role of ethnographic studies in empirical software engineering," IEEE Transactions on Software Engineering, vol.42, pp.786-804, 2016.

[2] Cosentino, V., Izquierdo, J. L. C., & Cabot, J, "A Systematic Mapping Study of Software Development With GitHub," IEEE Access, vol. 5, pp. 7173-7192, 2017

[3] DeFranco, J. F., &Laplante, P. A, "Review and Analysis of Software Development Team Communication Research," IEEE Transactions on Professional Communication, vol.60, pp. 165-182, 2017.

[4] Shahin, M., Babar, M. A., & Zhu, L, "Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," IEEE Access, vol. 5, pp. 3909-3943, 2017.

[5] Bate, I., Burns, A., & Davis, R. I, "An enhanced bailout protocol for mixed criticality embedded software," IEEE Transactions on Software Engineering, vol.43, pp.298-320, 2017.

[6] Fernández-Alemán, J. L., Carrillo-de-Gea, J. M., Meca, J. V., Ros, J. N., Toval, A., &Idri, A, "Effects of using requirements catalogs on effectiveness and productivity of requirements specification in a software project management course. IEEE Transactions on Education, vol.59, pp.105-118, 2014.

[7] Guillaume-Joseph, G., &Wasek, J. S, "Improving software project outcomes through predictive analytics: Part 2," IEEE Engineering Management Review, vol. 4, pp.39-49.

[8] Jorgensen, M., &Shepperd, M, "A systematic review of software development cost estimation studies," IEEE Transactions on software engineering, vol. 33, 2007.

[9] Sauer, C., Jeffery, D. R., Land, L., &Yetton, P, "The effectiveness of software development technical reviews: A behaviorally motivated program of research, "IEEE Transactions on Software Engineering, vol. 26, pp.1-14, 2000.

[10] NunamakerJr, J. F., Chen, M., &Purdin, T. D, "Systems development in information systems research," Journal of management information systems, vol.7, pp.89-106, 1990.

[11] Dyba, T., &Dingsøyr, T. , "Empirical studies of agile software development: A systematic review. Information and software technology, vol.50, pp.833-859, 2008.

[12] Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. "A decade of agile methodologies: Towards explaining agile software development," Elasveir, pp. 1213-1221, 2012.