Journal of Critical Reviews

ISSN- 2394-5125 Vol 7, Issue 8, 2020

Review Article

ADOPTIVE METHODS IN IDENTIFYING OF RISK FACTORS FOR AGILE PROCESS APPLICATIONS

[1]Nikhat Parveen, [2]P.Vidyulatha, [3]S.BalaVikas, [4]T.Viswanath, [5]S.Prathap

[1]Associate Professor, Department of computer science engineering,KoneruLakshmaiah Education Foundation, Vaddeswarami,Ap, India. E-mail: nikhat0891@gmail.com

[2]Associate Professor, Department of computer science engineering,KoneruLakshmaiah Education Foundation, Vaddeswarami,Ap, India. E-mail:latha22pellakuri@gmail.com

[3]Department of computer science engineering,KoneruLakshmaiah Education Foundation, Vaddeswarami,Ap, India. E-mail:balavikass999@gmail.com

[4]Department of computer science engineering,KoneruLakshmaiah Education Foundation, Vaddeswarami,Ap, India. E-mail:viswanadhviswa3@gmail.com

[5]Department of computer science engineering,KoneruLakshmaiah Education Foundation, Vaddeswarami,Ap, India. E-mail:prathapreddy1334@gmail.com

Received: 11.03.2020 *Revised: 12.04.2020* *Accepted: 28.05.2020*

ABSTRACT

The use of risk indicators can be helpful in making a risk assessment plan for any software project you plan to develop. This is important to streamline the process and help the manager make decisions. Despite the importance of risk management in software projects, outdated software developers are often to be overlooked. One reason for this fact is that the perception of risk is unreinforced and slanted, and its management does not see immediate practical results. For this perspective, this paper aims to identify, propose, and develop risk probability assessments that are specific to the aging software project environment in order to support the identification of risk elements. To achieve this goal, a list of risk indicators and risk elements has been developed with the help of a literature review, which is prepared for approval by an experienced program manager. The study was conducted to assess the likelihood of risks in rapid software.

Keywords: Agile software development(ASD), Risk probability, Waterfall model, Risk indicators, Risk elements

DOI: http://dx.doi.org/10.22159/jcr.07.01.01

1 INTRODUCTION

An Advanced Agilesoftware development is a much-liked space in the perspective of software engineering. His practice in relation to the changing nature of software logic has laid the groundwork for the latest project management methodologies supported in the software development environment. These methodologies are designed with one goal in mind: ways to achieve specific success results and to clarify the readings of these factors in the practical application of risk management. Because risk management [1] and [2] are the same in all aspects of project management, the use of aggressive software projects is becoming more and more important as the size and complexity of the project increases. In this context, we take into account the structural factors [3] and [4], but say that there is a need for indicators and measures that support aggressive software risk management within the framework of program project management beliefs and content. Identify risk factors and risk components at the time

In this paper, the inventory of risk elements for aggressive software development is known for its ability to develop software projects. The script was reviewed to accomplish this task. Risk components are collected from [6], [8], [11]. The list of known risk components for an aggressive project allows you to know what risks can be observed for this project and to look at the knowledge or criteria that may create that risk.

2 Agile Software Development

Agile software is used in projects where customer requirements change from time to time. This method is preferred over traditional software development methods when customer requirements change from time to time [5]. There are several types of fun software development. Agile has the ability to create new and respond to change if any.

Agile methods of software development continue to increase in popularity as firms are continually pursuing market speed and the ability to cope with changes in the current fast changing business environment[7].Agile approaches pose a practiceled push to build software in a high-speed and changing environment. Agile method advocates argue that Agile methods address many of the identified risks , particularly on the interface between software development and business organizations.

Factors such as inadequate coordination with users and stakeholders, lack of participation of users, failure to handle expectations of end users, misunderstanding of requirements and inability to accommodate ch anges in requirements and scope are claimed to be addressed positively by the implementation of Agile methods.

3 Waterfall Design

Waterfall design is a traditional way to develop software. Developers use this method if client requirements remain the same during software development. The cost for a waterfall model is lower than for a software development approach. However, there is a higher risk than ASD. The waterfall design time is longer compared to ASD.BACKGROUND

Test cases

We used a template consisting of several tests collected from various websites[9]. We also developed a method to compare the efficiency of software development with the effectiveness of traditional waterfall designs. We then tried to draw a graph with user input in both application development and waterfall models.

4 Proposed Model

This document offers the option to select the most efficient model based on customer requirements[10]. A number of other inputs, such as project deadlines, will be provided. Based on these inputs provided by the client or user, the proposed model compares the effectiveness of the aggressive design with that of the waterfall model. After predicting the efficiency of both models, place the graph. The ultimate goal of this model is to determine the risk factors and to determine whether or not to accept the test case. Finally, the client or customer decides which method is appropriate for their project and which method is most effective

A. Project Characteristics

Changing requirements will inevitably reduce project costs and stress periods. In this paper, we have identified a number of features and their positive and negative effects on software projects. We have identified four characteristics, each of which has positive and negative effects, and are shown in the table below .

Table 1 shows that the characteristics of the project may be detrimental to project management as well as to project team members. Properties that do not have a positive effect will always change.

SDLC is preferred in such situations. This is called the software development life cycle. The SDLC method is chosen depending on the needs and size of any project. The purpose of the SDLC is to provide a high-quality package that meets the customer's requirements. Typically, the SDLC is completed in the following steps

1. Comes to analyze needs
2. the need for shaping
3. Package design planning
4. Product development
5. Give it a try
6. Prepare for market and maintenance.

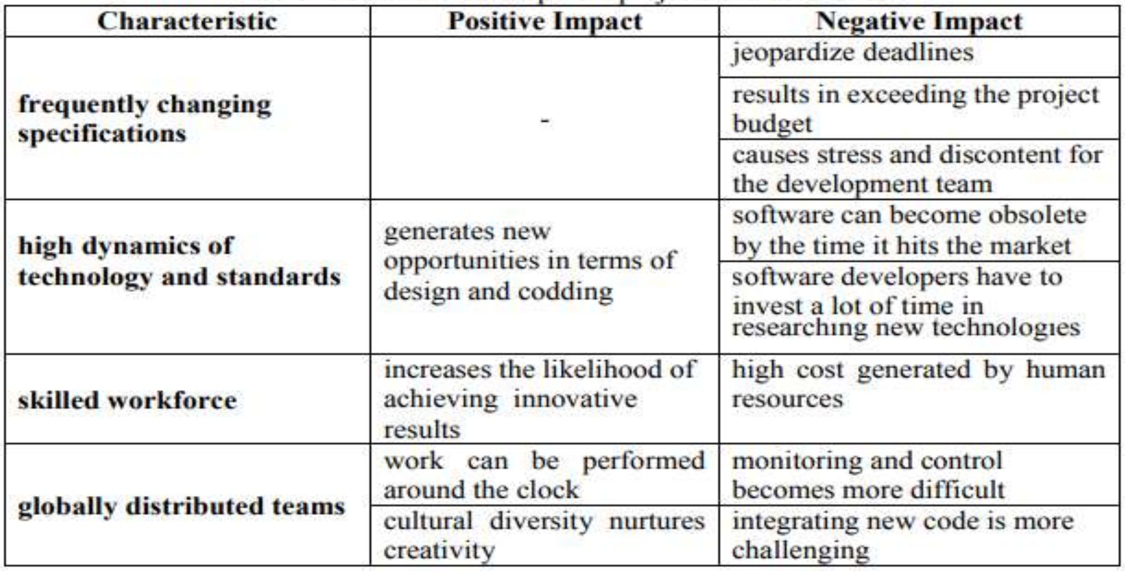
SDLC features:

1. The lowest cost
2. Flexibility and feedback
3. Parallel tasks

B. *.Waterfall Model*

The waterfall model can be a custom, downstream model that is often used in batch processing, so it dies and flows one by one as a result of all stages of research, design, production, implementation, construction, testing, and maintenance. like water down. This is the oldest and therefore the best of the SDLC.

Table 1. Software development projects characteristics



Characteristic	Positive Impact	Negative Impact
frequently changing specifications	-	jeopardize deadlines
		results in exceeding the project budget
		causes stress and discontent for the development team
high dynamics of technology and standards	generates new opportunities in terms of design and codding	software can become obsolete by the time it hits the market
		software developers have to invest a lot of time in researching new technologies
skilled workforce	increases the likelihood of achieving innovative results	high cost generated by human resources
globally distributed teams	work can be performed around the clock	monitoring and control becomes more difficult
	cultural diversity nurtures creativity	integrating new code is more challenging

HYDROELECTRIC POWER PHOTO:

1. The requirements are clear before development begins.
2. Each part is done at the time you give when you move on to the next step.
3. It can be a linear model and can be used directly.
4. The amount of resources required to use this model shall be nominal.

C. Agile Methodology

The Agile methodology is a repetitive process in which the client's requirements change over time. They change from time to time. It can be used quickly. You will need more than just the reputation of the product. It can be used where customers have the ability to change the scope of the project.

Advantages of the Agile method:

1. As a result of any step is not completed, we try to meet the requirements and keep it stable.

2. Documents are relatively small, which improves personal relevance.

3. Uncertainty between developers if the user is not cleaned, as it depends a lot on the interaction with the couturier.

D. Comparison of Agile and Waterfall Methods:

The waterfall model is called a step-by-step development model with a clearly defined delivery for each section. Several business interns conduct rigorous audits, and make sure that the project criteria are satisfactory before proceeding with the next part of the project. Flexible design is predicted in adaptive portfolio development strategies, but because it is versatile and transparent, it gives buyers rigid portfolio freedom. A detailed study of the table in support of some of the honorific options and the dynamic pattern is shown below

The Table 2 below shows how the design or features differ from the waterfall design to the aggressive design. There are several important differences between the traditional waterfall model and the aggressive software development model. This table discusses features such as the definition of requirements, understanding the requirements, and the total cost of the project. This draws a risk between the aggressive software development model and the waterfall model. The cost and guarantee of success of the project is low in the waterfall model, but very high in the aggressive model. One important thing that distinguishes the waterfall model from the aggressive design is that there is no continuous iteration of the requirements in the waterfall model. The waterfall design is also not flexible, but the flexible design is very flexible and the scope of the project can be changed at any time. Risk analysis is performed only on the waterfall model, but it is continuous on the aggressive model. These are the most important features that distinguish the waterfall model from the aging software model.

E. Method:

The proposed method seeks to compare and predict the most effective method to use in project development based on the client's or client's requirements. The client must fill in all required fields to find out which method to follow when using the software project development. There are several steps to the end result. These steps are described below.

Step 1: Username, email address, password, phone number, etc. must be registered.

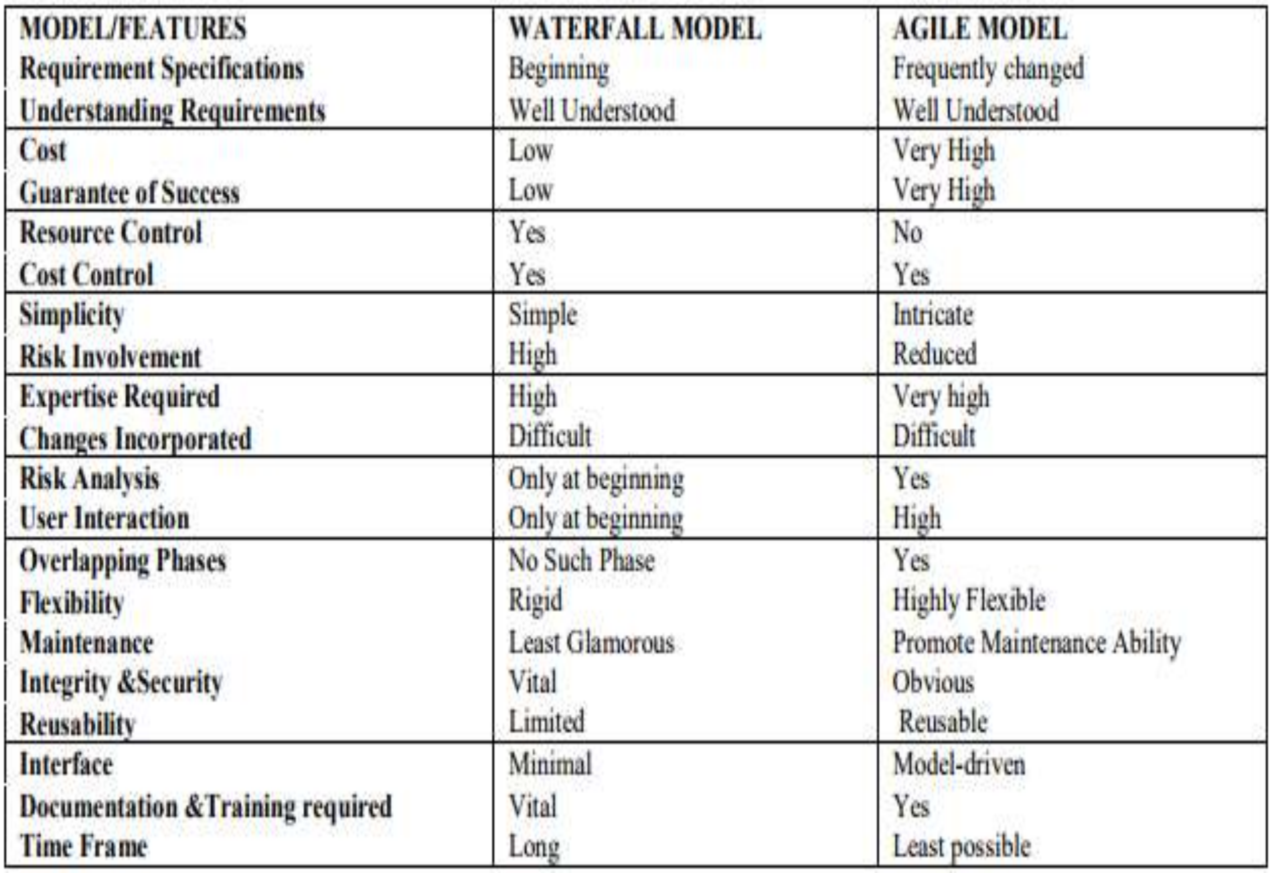
Step 2: After registration, the user will be redirected to the login page where they filled in their login name and password.

Step 3: If the information entered is correct, the user will be redirected to a test case template page consisting of several test cases.

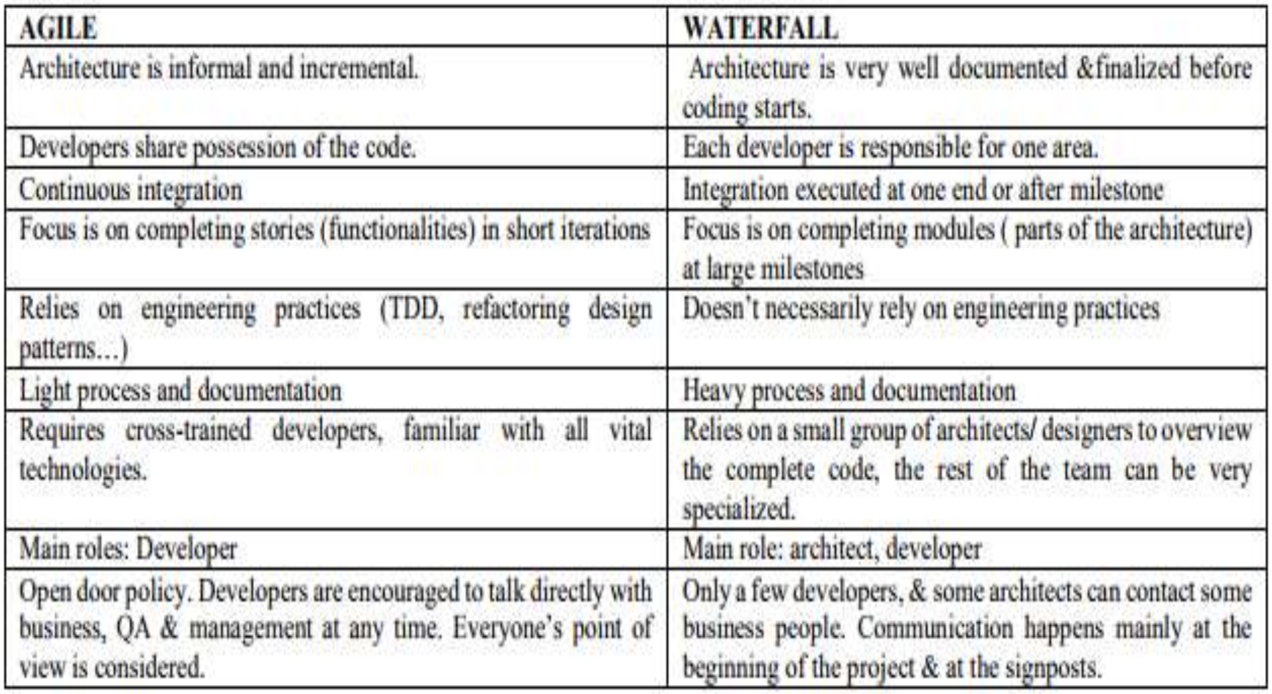
Step 4: After uploading the test template file, the test cases are checked and you can see if the uploaded test pages are past or failed or blocked.

Step 5: The user can check the forms and risks being analyzed in tabular form. The table shows the following important points.

1. The total number of test pages posted
2. Number of closed test cases
3. Number of test failures
4. Number of test runs
5. Test cost
6. Defect classification
7. Defect density, should never be negative. It also shows the repair time and detection time.



MODEL/FEATURES	WATERFALL MODEL	AGILE MODEL
Requirement Specifications	Beginning	Frequently changed
Understanding Requirements	Well Understood	Well Understood
Cost	Low	Very High
Guarantee of Success	Low	Very High
Resource Control	Yes	No
Cost Control	Yes	Yes
Simplicity	Simple	Intricate
Risk Involvement	High	Reduced
Expertise Required	High	Very high
Changes Incorporated	Difficult	Difficult
Risk Analysis	Only at beginning	Yes
User Interaction	Only at beginning	High
Overlapping Phases	No Such Phase	Yes
Flexibility	Rigid	Highly Flexible
Maintenance	Least Glamorous	Promote Maintenance Ability
Integrity &Security	Vital	Obvious
Reusability	Limited	Reusable
Interface	Minimal	Model-driven
Documentation &Training required	Vital	Yes
Time Frame	Long	Least possible



AGILE	WATERFALL
Architecture is informal and incremental.	Architecture is very well documented &finalized before coding starts.
Developers share possession of the code.	Each developer is responsible for one area.
Continuous integration	Integration executed at one end or after milestone
Focus is on completing stories (functionalities) in short iterations	Focus is on completing modules (parts of the architecture) at large milestones
Relies on engineering practices (TDD, refactoring design patterns…)	Doesn't necessarily rely on engineering practices
Light process and documentation	Heavy process and documentation
Requires cross-trained developers, familiar with all vital technologies.	Relies on a small group of architects/ designers to overview the complete code, the rest of the team can be very specialized.
Main roles: Developer	Main role: architect, developer
Open door policy. Developers are encouraged to talk directly with business, QA & management at any time. Everyone's point of view is considered.	Only a few developers, & some architects can contact some business people. Communication happens mainly at the beginning of the project & at the signposts.

Testing Metrics	Waterfall	Agile
Total No. of Testcases	8.0	8.0
Total No. of Blocked Testcases	2.0	2.0
Total No. of Failed Testcases	2.0	2.0
Total No. of Passed Testcases	4.0	4.0
Testcase Pass Rate	1.0	1.0
Defect Category	0.0	1.0
Defect Density	-0.75	0.25

Step 6: Upon that the user will be directed to a page where the user has to enter details regarding the deadlines and requirements of the project. The user must give manual inputs such as :

1. Requirements in days
2. Analysis in days
3. Design in days
4. Coding in days
5. Testing in days
6. Acceptance in days
7. Requirement in days

Step 7: After manually filling in all the above requirements, this template tries to graph all these requirements according to the level of accuracy. It advises users on which template to use more effectively during software development.

V. Results

The results are presented in the form of a graph shown in figure 3 Performance Analysis that meets the client's requirements and takes them into account. The graph shows the most effective method.

Consider the input provided by the client according to the based on the given forms and based on the project and end user requirements.

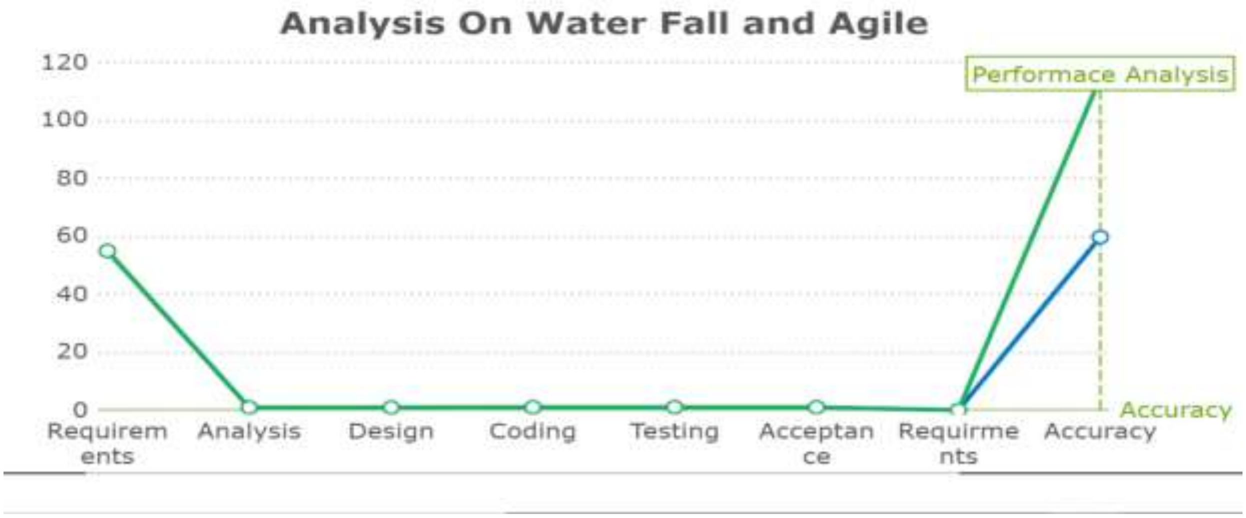


Figure 3. Performance Analysis

5 CONCLUSION

Conclusions and further work in this project have identified several risk elements and divided them into intelligent categories. Risk probability and performance analysis were performed to predict the mitigation of the risk. These are required for each software project under construction or under construction to reduce or avoid risk. In order to estimate the probability of a risk indicator in the future, it is necessary to determine the weight of the risk factor. Attitudes can be improved by incorporating knowledge and assisting in automated training to calculate risk effects.

REFERENCES

1. EmamHossain; Muhammad Ali Babar; Hyun-young Pike; June Werner, "Defining and Reducing the Risk of Using Scrum in Global Software Development: A Conceptual Framework," Software Engineering Conference, 2009. APSEC '09. Asia-Pacific, IEEE, 28 December 2009
2. EdzreenaEdzaOdzaly, Des Greer, Darryl Stewart, (2014)."Light Risk Management in Agile Projects". Pages 576-581. Presented at the 26th Software Engineering Conference in Vancouver, Canada
3. JuhaniIivari, NettaIivari,"The Relationship between Organizational Culture and Aggressive Placement,"Information and Software Technology 53 (2011) 509–520, Elsevier
4. Pellakuri Vidyullatha, Rajeswara Rao D, "Training and development ofartificial neural network models: Single layer feedforward and multi layer feedforward neural network", Journal of Theoretical and Applied Information TechnologyOpen Access, Volume 84, Issue 2, 20 February 2016, Pages 150-156
5. Julio Menezes Jr., Christine Gusmao, "Defining Risk Assessment Indicators ForSoftware Development Projects," Electronic Journal, pp. 1-24, 16, 1, p. 1, October. , 2013.
6. Marley M. Carvalho, "Critical Factors for the Success of the Six Sigma Project,"International Journal of Project Management, Volume 34, Elsever, August, November 2016, pp. 1505–1518.
7. RuchiAgrawal; Dipali Singh; Ashish Sharma, "Ranking and Optimizing Risk Factors for Fun Software Development," Ninth International Conference on Modern Computing (IC3), IEEE, March 20, 2017, pp. 1-7.
8. Pellakuri Vidyullatha, Rajeswara Rao D, "Knowledge based information mining on unemployed graduates data using statistical approaches", International Journal of Pharmacy and Technology, Volume 8, Issue 4, December 2016, Pages 21961-21966.
9. Parveen, N., Roy, A., SaiSandesh, D., SaiSrinivasulu, J.Y.P.R., Srikanth, N., "Human Computer Interaction Through Hand Gesture Recognition Technology ", International Journal of Scientific and Technology Research, Volume 9, Issue 4,April 2020,ISSN 2277-8616, pp:505-513.
10. Jammalamadaka, K., Parveen, N.,"Holistic Research of Software Testing and Challenges", International Journal of Innovative Technology and Exploring Engineering (IJITEE), ISSN: 2278-3075, Volume-8, Issue- 6S4, April 2019, PP: 1506-1521.
11. kumar S.A, Vidyullatha P "A comparative analysis of parallel and distributed FSM approaches on large-scale graph data " International Journal of Recent Technology and EngineeringOpen Access, Volume 7, Issue 6, April 2019, Pages 103-109.
12. S. Velliangiri, P. Karthikeyan & V. Vinoth Kumar (2020) Detection of distributed denial of service attack in cloud computing using the optimization-based deep networks, Journal of Experimental & Theoretical Artificial Intelligence, DOI: 10.1080/0952813X.2020.1744196
13. Praveen Sundar, P.V., Ranjith, D., Vinoth Kumar, V. et al. Low power area efficient adaptive FIR filter for hearing aids using distributed arithmetic architecture. Int J Speech Technol (2020). https://doi.org/10.1007/s10772-020-09686-y
14. Vinoth Kumar V, Karthikeyan T, Praveen Sundar P V, Magesh G, Balajee J.M. (2020). A Quantum Approach in LiFi Security using Quantum Key Distribution. International Journal of Advanced Science and Technology, 29(6s), 2345-2354
15. Karthikeyan, T., Sekaran, K., Ranjith, D., Vinoth kumar, V., Balajee, J.M. (2019) "Personalized Content Extraction and Text Classification Using Effective Web Scraping Techniques", International Journal of Web Portals (IJWP), 11(2), pp.41-52
16. Vinoth Kumar, V., Arvind, K.S., Umamaheswaran, S., Suganya, K.S (2019), "Hierarchal Trust Certificate Distribution using Distributed CA in MANET", International Journal of Innovative Technology and Exploring Engineering, 8(10), pp. 2521-2524