ISSN 2348 – 8034
Impact Factor- 3.155

GLOBAL JOURNAL OF ENGINEERING SCIENCE AND RESEARCHES

A SURVEY ON QUALITY SOFTWARE PROJECT MANAGEMENT SYSTEM FOR DEFECT ANALYSIS

Tahira Mahboob*[1], Humaraia Abdual Ghafoor*[2] and Warda Imtiaz[3]
[1]Assistant Prof. Department of Software Engineering, Fatima Jinnah Women University, Pakistan
[2,3]Department of Software Engineering, Fatima Jinnah Women University, Pakistan

ABSTRACT

This paper is a survey on quality Analysis for Software Project Management System and it elaborates that one of the advantages of defect prevention is the beneficial effect it may have on interval: at the start of process, good quality results in low defects ratio to be found and corrected in the later phases of the process, thus causing an indirect reduction of intervals. As there are three inter-dependent factors that controls project's development processes; interval, quality and cost. The market pressures continue to demand agility in new features; the challenge to project management is to meet those demands while not sacrificing the quality. The root-cause analysis is the core of this threefold study. . It is necessary to point out the defects from the given set of projects at the first step, for enhancing the quality of the software process; then it involves classification and analysis of the pattern and further it involves deduction for defects prevention. This paper discusses the role of defect analysis as a feedback mechanism to progress the quality and production in an iteratively developed software project.

Keywords: *Defect Analysis, Defect Prevention, ODC, Defect Trigger, Defect Injection Metric, Defect Severity, evaluation of defects, Automatic Defect Classification, Root Cause Defect Analysis.*

I. INTRODUCTION

The failure analysis and root-cause analysis in integration are potentially more valuable than subjective assessments, because they quantify defect costs for a specific organization. Software defect data is most important available management information source, for software process improvement decisions. The researchers were looking for conducts on the other hand; to guide defect correction effort, using information retrieval and natural language processing, to cluster defect reports. Right now, there is no estimation method grew through exact investigation to assess the choice capacity of a task chief towards asset designation for viable deformity administration. The imperfection system is taking into account examining the imperfections that had risen up out of different phases of programming advancement like Requirements, Design, Coding, Testing and Timeline; surrenders because of absence of time amid improvement. The deformity infusion metric quality, once computed, serves as a measuring stick to make an examination in the upgrades made in the product process improvement between comparable arrangements of ventures. Significant disarray wins at the last phases of programming advancement as to which desert found has a place with which period of programming improvement life cycle. Systems like main driver investigation and orthogonal imperfection arrangement are a portion of the normally utilized practices.

II. SOFTWARE PROJECT MANAGEMENT SYSTEM TECHNIQUES

2.1)Classification and evaluation of defects in a project retrospective

One of the advantages of defect prevention is the beneficial effect it may have on interval: at the start of process, good quality results in low defects ratio to be found and corrected in the later phases of the process, thus causing an indirect reduction of intervals. As there are three inter-dependent factors that controls project's development processes; interval, quality and cost. The market pressures continue to demand agility in new features; the challenge to project management is to meet those demands while not sacrificing the quality. The root-cause analysis is the core of this threefold study. This paper discuss the novel approach that have been used for defect and root cause classification, in addition to that the mechanisms used for randomly selecting the MRs; to analyze analyses via a web interface. The paper presents the results of analyses of the MRs and explains the defects and root causes that were found. Then the researcher concludes the report on the root-cause analysis. The project context of this study is a network element: a flexibly configurable transmission system in an optical network. The limitations of this paper are: the root-cause study focuses on defect analysis and on determining the underlying root causes of the defects.

2.2) Defect Handling In Software Metrics

ISSN 2348 – 8034
Impact Factor- 3.155

In various software projects defect handling is one of the dominant and important activities involved. Software projects will make defect removal and prevention much easier in case of earlier defect identification. It is necessary to point out the defects from the given set of projects at the first step, for enhancing the quality of the software process; then it involves classification and analysis of the pattern and further it involves deduction for defects prevention. The experiences from the past are usually taken by software engineers to prevent the defects from re-occurring. The undesirable aspects of the software quality are generally represented as a defect. In this research paper the study about the various types of defect prevention techniques is analyzed, and then researchers have undergone through the survey of a two-step software defect prediction model for improving the quality of software: COQUALMO cost constructive model. Better software results are produced when early identification of the defect is introduced.

2.3)Software Failure Analysis for High-Return Process Improvement Decisions

To determine the weaknesses in development processes and to decide what changes are needed; the valuable tools in enabling organizations have become: the software failure analysis and root-cause analysis. Transfer process learning from individuals to organizations is a useful way to evaluate software defect. It includes analyzing software defects, also brainstorming the root causes of those defects and incorporating what is learned during training and process changes; so that the defects never occur again. The failure analysis and root-cause analysis in integration are potentially more valuable than subjective assessments, because they quantify defect costs for a specific organization. Software defect data is most important available management information source, for software process improvement decisions. This paper elaborates the aspect that how software defects data is influential management in sequence source. Using it effectively will help attain an optimal balance between respond to defect information, and proactively taking decisions toward thwarting future defects.

2.4) Automatic Defect Classification: An Experience Applying Natural Language Processing

This paper shows early results of a research effort that merges two approaches on Quality Management; on one hand, working with process development initiatives involving defect underlying analysis, and the problem of measuring inter-rater trustworthiness for defect classification taxonomy has occurred in a software development organization. The researchers were looking for conducts on the other hand; to guide defect correction effort, using information retrieval and natural language processing, to cluster defect reports. The value of the fact that a software development organization can improve from their defects nevertheless, the analysis of each individual defect can be tiresome, error prone, and time taking, has already been established by previous research. As a proxy to estimate remaining defect density in delivered software a practical application can also include the use of defect clustering combined with contextual information from the organization. As mentioned earlier, clustering can also be used to prioritize which defect to cater first.

2.5) Using Defect Analysis Feedback for Improving Quality and Productivity In Iterative Software Development

The preferred approaches for developing software avoiding some of the problems of the waterfall model that are better suited for the rapid changing world are now iterative development models. The software is developed in a series of iterations, each iteration producing a working system; which is then used to build the next more capabilities version of the system, in an iterative development. This paper discusses the role of defect analysis as a feedback mechanism to progress the quality and production in an iteratively developed software project. The researchers discuss how investigation of defects establish in one iteration can produce the feedback for defect prevention in upcoming iterations, leading to the improvement quality and productivity. A few checkpoints are made at which investigation is carried out and results sustained in a waterfall sort model. In long running tasks, such an investigation could be possible at normal periods. At Info-framework we have seen profits of applying this procedure in these tasks also.

2.6) An Analytical Approach for Project Managers in Effective Defect Management in Software Process

Deformity estimation and expectation is a portion of the primary regulating components for the accomplishment of programming tasks in any product industry. Development and competency of a task supervisor in proficient forecast and estimation of asset capacities are one of the vital main impetuses towards the era of great programming. Right

ISSN 2348 – 8034
Impact Factor- 3.155

now, there is no estimation method grew through exact investigation to assess the choice capacity of a task chief towards asset designation for viable deformity administration. This paper draws out an exact study did in an item based programming association. Our profound examination on a few ventures tosses light on the effect of choice ability of venture director towards achievement of a previously stated goal. The paper empowers venture administrators to increase further mindfulness towards the hugeness of prescient situating in asset allotment keeping in mind the end goal to grow brilliant deformity free programming items. It likewise upgrades the development level of the organization and its steadiness in the focused air. The advancement and utilization of programming keep on being one the key innovations for the accomplishment in any business area. Route back a large portion of a century, nobody could have anticipated that the product would turn into a key innovation for business, science and designing.

2.7)Modelling and Analysing of Software Defect Prevention Using ODC

As the time passes the product intricacy is expanding and because of this product unwavering quality and quality will be influenced. Furthermore, for measuring the product dependability and quality different imperfection estimation and deformity following instrument will be utilized .Software imperfection anticipation work normally concentrates on individual examination and testing system. ODC is a component by which we abuse programming desert that happen amid the product improvement life cycle. Orthogonal deformity arrangement is an idea which empowers engineers, quality directors and task chiefs to assess the viability and rightness of the product. Programming imperfection avoidance is a vital piece of the product advancement. The quality; dependability and the expense of the product item intensely rely on upon the product imperfection location and counteractive action process. In the advancement of programming item 40% or a greater amount of the task time is spent on deformity discovery exercises. Programming deformity anticipation exploration has proposed new investigation and testing strategies and has concentrated on and analyzed diverse assessment and testing systems. In the paper the fundamental thought is to give execution of ODC in certifiable application. It starts with and review of different deformity arrangement plans took after by ODC ideas. The last part will depict how we will embrace ODC in programming improvement. The end of this paper portrays the Improvement in programming venture in the wake of executing ODC. Page Layout a simple approach to agree to the meeting paper designing necessities is to utilize this report as a format and basically sort your content into it.

2.8) Defect prevention based on 5 dimensions of defect origin

The expanding reliance of society on programming and the critical results that a disappointment can result in obliges the need to discover the deformities at the beginning itself. In view of the lessons learnt from the prior arrangement of activities, a deformity structure highlighting the 5 Dimensions (Ds) of imperfection starting point is proposed in this work. The imperfection system is taking into account examining the imperfections that had risen up out of different phases of programming advancement like Requirements, Design, Coding, Testing and Timeline; surrenders because of absence of time amid improvement. This study is not constrained to simply recognizing the inception of imperfections at different periods of programming advancement additionally figures out the explanations behind such surrenders, and deformity preventive (DP) measures are proposed for every sort of imperfection. This work can help specialists pick compelling deformity shirking measures. Notwithstanding touching base at imperfection structure, this work additionally proposes a deformity infusion metric taking into account seriousness of the deformity instead of simply surrender check, which gives the quantity of balanced deformities created by an undertaking at different stages. The deformity infusion metric quality, once computed, serves as a measuring stick to make an examination in the upgrades made in the product process improvement between comparable arrangements of ventures. Significant disarray wins at the last phases of programming advancement as to which desert found has a place with which period of programming improvement life cycle. Systems like main driver investigation and orthogonal imperfection arrangement are a portion of the normally utilized practices.

2.9) Establishing a Defect Management Process Model for Software Quality Improvement

Software is developed by human so defect will be remaining in whole life of software. Quality software with the least number of defects to decrease the influence of problems in the organization is basically the main goal of well organized software defect management process. There are three levels of defect management process which are defect detections, defect analysis and defect prevention .Defects are identified and fixed at first level. In second level find the root causes of the defect why they are not detected in first level. And the third level is related with the defect

prevention from recurring in the future. The main research is that how to produce quality software with minimum number of defects. The main factor of success is the software will be in time and budget. In this research paper, establish a defect management process model that reduce number of defects and provide a quality software product. Mostly Organizations are using ITIL defect management process model. The major challenges in the ITIL defect management process model are lack of performance and less participation of customer in process.

2.10) Better Management of Defects for Improving Software Processes

Testing is very important after developed software because no software can build defect free software. Defect Tracking System is the tool that used for reporting defects after testing software. Software quality enhanced by defects reported. In this research paper research is on how defects are managed and approaches that are used for managing defects and for improvement that will prevent the defects in future. Defect is basically erroneous information in software. This erroneous information is due to an error in design, specification and mostly in requirement. These defects and errors are finding in testing and identified during reviews because these defects creates complexity in software. Defect defined as it is unplanned event that occurs during reviews and in testing. These defects require investigation and correction and these are created when results are different from expected results. When defect is indentified then its related information is recorded and this information is called Defect report .Developer review the defect report and try to resolve.

2.11) Myths and Strategies of Defect Causal Analysis

Causal analysis activities are Sigma, CMMI, and Lean and these are popular process improvement approaches. The concept of causality technique is used in causal analysis which is misunderstood and misapplied. For defect causal analysis many different processes, tools, and techniques are developed now-a- days. These different processes, tools, and techniques are successful in many situations. Many Organizations invest large amounts in software and training for deploy them. The application of these techniques provides sources of problems. For correct identification and effective decision of “deep” problems, a checklist implementation of the techniques is not enough. In the software industry it is becoming more common and focuses on increasing quality improvement. It is important training attentive planning is used to ensure for successful implementation. This research paper finds the major problems and suggests the solution of these problems.

2.12) Contemporary Trends in Defect Prevention a Survey Report

Due to different type of defects many software projects fails to meet require level of quality and standards. During the course of requirement solicitation, designing and development defects introduced .These defects inevitably delay the protected deployment or effective operations of the software systems. The lack of proper defect prevention planning for formulating the software architecture is the One of the key reason. Defect prevention must be the critical phase because it has influence on quality of the product which cannot be help. This research paper includes different defect prevention techniques and analyses these techniques critically. Defect prevention is improving quality of software product. Within the available budget and time the focus of techniques is to develop improved quality products. Quality of a product associate with the number of defects because minimal defects will lead to achieve enhanced quality.

This paper is ordered into four sections. A brief introduction of the defect prevention and its classification is in first section. The literature review and summarizes the related work in second section. In third and fourth sections conclusion and future work discuss.

ISSN 2348 – 8034
Impact Factor- 3.155

2.13) A Case Study in Root Cause Defect Analysis

Interval, quality and cost are three interdependent factors that make our software development processes. New features rapidly demand in market without any defect and with high quality. An upstream quality improvement Practice is one advantage of defect prevention. Indirect interval reduction produced by higher quality early in the process results in fewer defects to be found .the defect Modification Requests (MRs) revealed by root cause defect analysis study. In this research paper defect and root cause classification and the mechanisms and used for arbitrarily selecting the MRs to examine and collecting the analyses via a web interface. 5-10 people teams in the NE software are developed.NE configuration consists of different hardware board types and equal to 150 different software components. Subsystem is the architectural unit that a software team is responsible for a collection of functionally related components, which altogether form an architectural unit. The mission of the RCA project was find systematic root causes of defects, during the maintenance release analyze major customer reported MRs, reduce number of critical defects in project.

2.14) Defect Analysis and Prevention for Software Process Quality Improvement

"An ounce of aversion is justified regardless of a pound of cure." In programming, these interpretations interpret into the normal perception that the longer a deformity stays in process, the more lavish it is to alter besides programming deformities are extravagant and time expending. The expense of discovering and adjusting imperfections speaks to a standout amongst the most lavish programming improvement exercises. What's more, that as well, if the slips escape till the last acknowledgement testing phase of the venture life cycle, then the undertaking is at a more noteworthy hazard regarding now is the ideal time and Cost variables. A little measure of exertion spent on quality affirmation will see great measure of expense reserve funds in terms of distinguishing and dispensing with the imperfections. To pick up a more profound comprehension of the adequacy of the product process, it is fundamental to look at the points of interest of imperfections distinguished in the past undertakings and to study how the same can be dispensed with due to process enhancements and more up to date approaches. This paper will concentrate on discovering the aggregate number of imperfections that has happened in the product improvement process for five comparable ventures and goes for arranging different deformities utilizing first level of Orthogonal Deformity Classification (ODC), discovering underlying drivers of the deformities and utilize the learning of the tasks as preventive thoughts. The paper too showcases on how the preventive thoughts are executed in another set of ventures bringing about the decrease of the quantity of comparable deformities.

2.15) Difficulties in Managing Software Problems and Defects

Numerous IT associations are battling with the expanding number of programming issues furthermore, deserts. The quantity of programming issues and deformities has expanded because of complex IT frameworks, new innovations, and tight venture plans. Programming quality issues can quickly expand the expenses of programming support and advancement. Sadly, bolster groups of IT associations have constrained assets for determining programming issues and imperfections. Regularly, they don't have decently characterized methodology models for issue administration. Furthermore, conventional imperfection administration models are most certainly not sufficient to administration arranged programming business in which issue determination requires correspondence between a few administration suppliers. The methodology of overseeing issues and deformities incorporates an expansive number of challenges also, challenges however has been given little thought in programming building research. The exploration work of this proposition is sorted out around four objectives. The primary objective is to study programming quality certification routines that can be utilized to distinguish imperfections. The second objective is to distinguish the challenges that associations have in overseeing programming deformities and which upgrades are expected to existing imperfection administration models. The third objective is to study the ideas of administration situated issue

ISSN 2348 – 8034
Impact Factor- 3.155

administration. The fourth objective is to study difficulties and troubles of administration arranged issue administration techniques.

2.16) Handling of Software Quality Defects in Agile Software Development

Programming quality affirmation is concerned with the effective and powerful advancement of vast, dependable, also, top notch programming frameworks. In coordinated programming advancement and support, refactoring is a vital stage for the constant change of a product framework by uprooting quality deformities like code scents. As time is critical figure coordinated advancement, not all quality deformities can be evacuated in one refactoring stage (esp. in one emphasis). Documentation of value surrenders that are found amid robotized or manual revelation exercises (e.g., pair writing computer programs) is important to evade exercise in futility by rediscovering them in later stages. Lamentably, the documentation and treatment of existing quality deserts and refactoring exercises is a typical issue in programming support. To review the reasons why changes were done, data must be separated from either exclusive documentations or programming forming frameworks. In this part, we depict a procedure for the repeating what's more, practical disclosure, taking care of, and treatment of value deserts in programming frameworks. An annotation dialect is exhibited that is utilized to store data about quality deformities found in source code and that speaks to the deformity furthermore, treatment history of a piece of a product framework.

III. ANALYSIS DESCRIPTION

Results of analysis of evaluation parameters are shown in the table 2.

Sixteen techniques are discussed against 20 parameters. Important parameters are discussed in the first half and rest of them is discussed in the later. In Table 1 all the parameters along with their meaning and possible values are presented while table 2 shows the existence of these parameters in respective search paper techniques and related with quality. Cost effective, performance, efficiency, reliability, maintainability, scalability, effortless and case study are discussed. Almost all mentioned techniques are cost effective and have good performance abilities. Performing activities/tasks without any failure is reliability in table 2. All authors discuss reliability factor in research in table 3.New features can be easily adapted and are operable by the existing system or techniques is robustness shows in table 2.In Analysis table description there is no robustness factor. Complexity is basically to what extent modules are inter-related and dependent in table 2. Provide the techniques which are not fullfil the requirements for remove the complexity in table 2. Performance, Efficiency, Integrity. Reusability, Security are the other parameters describe in table 2 and analysis is shown in table 2. Case study is the basically reference to some experimentation in table 1. T.HOFMANN, J. G. Dy and C. E. Brodley, L. K. Saul and S. T. Roweis, Q. V. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng research not provide the information of parameter “case study” which is analyze by table 3 evaluations. Datasets are unbalanced in information retrieval is the precision in table 1. Accuracy of a product shows that there is number of correct predictions.

IV. CONCLUSION

This research paper includes different defect prevention techniques and analyses these techniques critically. Defect defined as it is unplanned event that occurs during reviews and in testing. These defects require investigation and correction and these are created when results are different from expected results. The project context of this study is a network element: a flexibly configurable transmission system in an optical network. Mostly Organizations are using ITIL defect management process model. The major challenges in the ITIL defect management process model are lack of performance and less participation of customer in process.The limitations of this paper are: the root-cause study focuses on defect analysis and on determining the underlying root causes of the defects. In the software industry it is becoming more common and focuses on increasing quality improvement. It is important training attentive planning is used to ensure for successful implementation. This research paper finds the major problems and suggests the solution of these problems.

REFERENCES

1. *M. Leszaka, D E. Perryb, D. Stolla. "Classification and evaluation of defects in a project retrospective." The Journal of Systems and Software Volume 61, (2014). http://users.ece.utexas.edu/~perry/work/papers/DP-02-jss.pdf*
2. *[2]-Mittal1, S K. Dubey. "Defect Handling In Software Metrics." ISSN : 2278 – 1021 International Journal of Advanced Research in Computer and Communication Engineering Vol. 1, (May 2012). http://www.ijarcce.com/upload/may/Defect%20Handling%20Using%20COQUALMO%20Defect%20Prediction%20Model.pdf*
3. *R B. Grady. "Software Failure Analysis for High-Return Process Improvement Decisions." Hewlett-Packard Journal August 1996. Article 2 http://www.hpl.hp.com/hpjournal/96aug/aug96a2.pdf*
4. *S. Matalonga, T S. Feliu, V. Rus. "Automatic Defect Classification: An Experience Applying Natural Language Processing." Dunn Hall, Memphis TN. Department of Computer Science, University of Memphi.(2011) http://fi.ort.edu.uy/innovaportal/file/2038/1/automaticdefectclasification.pdf*
5. *P. Jalote. "Using Defect Analysis Feedback For Improving Quality And Productivity In Iterative Software Development." Department of Computer Science and Engineering Indian Institute of Technology Kanpur. (2010) http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1609661&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D1609661*
6. *T R G. Nair, V. Suma, N R S. Kumar. "An Analytical Approach for Project Managers in Effective Defect Management in Software Process." 2011 5th Malaysian Conference in Software Engineering (MySEC). http://arxiv.org/ftp/arxiv/papers/1203/1203.6439.pdf*
7. *p. Trivedi, S. Pachori. "Modelling and Analysing of Software Defect Prevention Using ODC." International Journal of Software Engineering & Applications(IJSEA), Vol.3, No.4, (July 2012) http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.205.9265*
8. *S. Kumaresh, B. Ramachandran. "Defect Prevention Based On 5 Dimensions Of Defect Origin" (2013) http://airccse.org/journal/ijsea/papers/3412ijsea07.pdf*
9. *H. A. Khan. "Establishing a Defect Management Process Model for Software Quality Improvement." Internet: www.ijfcc.org/papers/232-B10100.pdf, 6, December 2013.*
10. *S.Mittal1 , K.Solanki2 , A. Saroha3. "Better Management of Defects for Improving Software Processes." Internet: journaldatabase.info/download/pdf/better_management_defects_for Improving Software Processes.pdf, 02, Aug 2011.*
11. *D.N.Card. "Myths and Strategies of Defect Causal Analysis." Internet: www.researchgate.net/...Myths_and_strategies_of_defect_causal_analysis.pdf, 22 October 2006.*
12. *M.Faizan, M. N. Ahmed Khan. "Contemporary Trends in Defect Prevention a Survey Report." Internet: www.researchgate.net/.../09e415141ce5eecbd3000000.pdf, 2, March 2012.*
13. *M.Leszak, D.E.Perry & D.Stoll. "A Case Study in Root Cause Defect Analysis." Internet: users.ece.utexas.edu/~perry/work/papers/DP-00-icse-rca.pdf, 7, Nov 2011.*
14. *S. Kumaresh & R. Baskaran. "Defect Analysis and Prevention for Software Process Quality Improvement." Internet: ijcaonline.org/volume8/number7/pxc3871759.pdf, .7, October 2010.*
15. *M. JÄNTTI. "Difficulties in Managing Software Problems and Defects." Internet: epublications.uef.fi/pub/urn_isbn.../urn_isbn_978-951-27-0109-4.pdf, 1st February 2008.*
16. *J.RECH. "Handling of Software Quality Defects in Agile Software Development." Internet: joerg rech.com/Paper/BC_Rech_AgileQDH_final.pdf , Oct. 25, 2008.*

TABLE 1 PARAMETERS, MEANINGS POSSIBLE VALUES

S#	*Quality Parameters*	*Meanings*	*Possible Values*
1.	***Security***	Ability of a system that no one can access the personal information of a specific person	Yes, No, Not defined
2.	***Automation***	Everything or the huge module of the system is fully	Yes, No, Not defined

ISSN 2348 – 8034
Impact Factor- 3.155

		automatic or can do work on its own	
3.	***Cost effective***	Effective or productive process in relation to its cost.	Yes, No
4.	***Extendibility***	A system that is easy to adapt to new specification.	Yes, No
5.	***Performance***	Backtracking or recovery process defines the performance of the system	Yes, No
6.	***Integrity***	How well the software protects its programs and data against unauthorized access	Yes, No
7.	***Reusability***	Existing techniques can be reused to form other applications	Yes, No
8.	***Robustness***	Appropriate performance of a system under cases not covered by the specification. This is complementary to correctness.	Yes, No
9.	***Scalability***	Capability to cope up and perform under an increased or expanding workload.	Yes, No
10.	***Efficiency***	Low utilization of resources, lower response time and mean time of failure and recovery define the performance of the system.	Yes, No
11.	***Diversity***	How much a system can perform in the changeable environment/ user response	Yes, No, Not defined
12.	***Portability***	The ease of installing the software product on different hardware and software platforms	Yes, No
13.	***Ease of use***	The ease with which people of various backgrounds can learn and use the software.	Yes, No
14.	***Reliability***	Ability of a program to perfume its functions without experiencing failure.	Yes, No
15.	***Virtualization***	To how much extant the paper work is digitalized	Yes, No
16.	***Maintainability***	The ease of changing the software to correct defects or meet new requirements	Yes, No
17.	***Compatibility***	Software that is composed of elements that can easily combine with other elements.	Yes, No
18.	***Effortless***	The quality of a system that makes the user to use it easily	Yes, No
19.	***Complexity***	Too large or too complicated to be solved in house infrastructure	Yes, No
20.	***Case study***	Support with examples	Yes, No

Note: ***Y =yes, N=no, and ND= not defined.***

TABLE 2: ANALYSIS TABLE OF PARAMETERS AGAINST AUTHORS

S#	*Author*	*Security*	*Automation*	*Cost effective*	*Extendibility*	*Performance*	*Integrity*	*Reusability*	*Robustness*	*Scalability*	*Efficiency*	*Diversity*	*Portability*	*Ease of use*	*Reliability*	*Virtualization*	*Maintainability*	*Compatibility*	*Effortless*	*Complexity*	*Case study*
1	***M. Leszaka, D E. Perryb, D. Stolla***	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	ND	Y	Y	Y	Y	Y	Y	Y	N

ISSN 2348 – 8034
Impact Factor- 3.155

2	***Mittal1, S K. Dubey***	Y	Y	N	Y	Y	Y	N	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
3	***R B. Grady***	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	Y	ND
4	***S. Matalonga, T S. Feliu, V. Rus***	N	ND	Y	N	Y	Y	N	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
5	***P. Jalote***	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y	Y	Y	Y	N	Y
6	***T R G. Nair, V. Suma, N R S. Kumar***	Y	Y	N	N	Y	N	N	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y
7	***P. Trivedi, S. Pachori.***	Y	Y	N	Y	N	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	Y	Y	N
8	***S.Kumaresh,B.Rama chandran***	Y	Y	Y	Y	Y	N	Y	N	N	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	ND
9	***H. A. Khan***	Y	Y	Y	Y	N	Y	Y	Y	Y	N	N	ND	Y	Y	Y	Y	Y	Y	Y	N
10	***S.Mittal , K.Solanki,A.Saroha***	Y	Y	N	N	Y	Y	N	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y
11	***D.N.Card***	Y	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	Y	ND
12	***M.Faizan, M.N. Ahmed Khan***	N	ND	Y	N	Y	Y	N	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	Y
13	***M.Leszak, D.E.Perry & D.Stoll***	Y	Y	Y	Y	Y	Y	Y	N	Y	N	Y	Y	Y	N	Y	Y	Y	Y	N	Y
14	***S.Kumaresh &R.Baskaran***	Y	Y	N	N	Y	N	N	Y	N	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y
15	***M. JÄNTTI***	Y	Y	N	Y	N	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	N	Y	N	N
16	***J.RECH***	Y	Y	Y	Y	Y	N	Y	N	N	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	ND