Towards a Framework Empowering Software Testing Process for Better Quality

Osama M. Abu Elnasr
College of Computers and Information Sciences, Egypt.

Magdy Z. Rashad
College of Computers and Information Sciences, Egypt.

Mohammed A. Abo-Elsoud
College of Computers and Information Sciences, Egypt.

Gamal M. Aly
College of Engineering, Egypt.

ABSTRACT
Software development frameworks are considered the engine that leads the development activities within software development organizations, so the main interest in this research is establishing a framework to effectively lead the development activities to enhance the software quality. The proposed framework relates each development phase with its available artifacts, software metrics can be extracted from its artifacts and relates the test practices and QA best practices during the SDLC. It provides an organized approach to handle miscommunication and poor team management that usually lead to team confusion. It also provides a way to disengage the ambiguity between the software fault and failure through establishing a clear relationship between the both concepts and developing automated models to predict the software correctness and measure its reliability.

Keywords
SDLC, QA, Static Analysis, Testing, Fault, Failure, Correctness, Reliability.

1. INTRODUCTION
Software development organizations seek to improve the quality of their software by continuous working to adapt their development models. Some still depend entirely on the traditional model and others moving strongly towards the agile methodologies with closed eyes about the advantages and disadvantages of the both models, despite their influence on the pillars of the software development process. There are great attempts in the research environment to bring the views and to create models combine the advantages of each model [1-5], other attempts seek to automate the verification activities in order to get a complete control of the behavior of the software and its related quality attributes [6-10].

Software measurement analysis, fault prediction and failure estimation techniques become rich area of research related to verification activities. By evaluating the attributes of the software, we can know its status, characteristics and behavior, [11,12]. The fault proneness [13-16], defect density [16-19], and failure radiation information [20] provide important guidelines to testing practitioners to prioritize their testing effort and assign verification and validation activities.

Most studies related to the research area are considered individual contributions that seek to automates some of the software development activities without concern for creating a framework that includes those contributions to building an integrated framework can be relied upon to manage the three elements of the development system; people, process and product [21-25]. The paper extends the work done by Yu BengLeau [1] and Milad Hanna [10]. Yu BengLeau [1] has been introduced a review paper that explains the advantages and disadvantages of both traditional and agile methodologies and suggests improvements for current agile development for enhancing organizational project management. Milad Hanna [10] has been introduced a new scripting technique that facilitates the process of automating the execution phase through software testing in an industrial context.

The paper seeks to provide a framework that manages the stakeholders and draws the relationship between each process within the various phases of the system and the deliverables of them with clarifying entity responsible for the production and the beneficiary of its existence in order to structure the inner workings of the system. It sheds light on the pivotal role played by the applications of artificial intelligence to automate the software fault prediction process in a manner of errors spread, their location, and the extent of their impact on the development process and their related quality attributes. It also gives a broad interest to study the behavior of the software product through analyzing and reviewing the deliverables of each phase using what are known by the term software metrics which are used as a basis to build a system capable of monitoring the correctness of software product and its reliability.

The remainder of this paper is organized as follows; Section 2 highlights the problems associated with current area of research. Section 3 provides the full description of the proposed framework. Section 4 introduces the conclusion and the future work.

2. PROBLEM DESCRIPTION
In spite of the successive efforts to develop models of development, but it still suffers from foggy specification of the basic development components of the system ; people, process and product. There are still issues concerning the management of stakeholders, not to mention chaotic the overlapping among the development processes and their integration with each other to produce cost effective qualified software product, and other drawbacks such as:

- Lack of a conceptual framework that provide a clear distinction between software fault and failure, their causes, effects and their prediction and estimation models.
- Lack of an organized approach to handle miscommunication and poor team management that usually lead to team confusion.
- Lack of the distinction between static analysis activities and dynamic testing activities and their scope in handling software quality.

- Lack of a conceptual specification of the tasks of each development process, its interest, and its deliverables.
- Fogginess to obtain information about the progress of the development process until the late stages of the age of the system.

It becomes very important to develop a conceptual framework that brings the views between traditional development activities, agile practices and their related quality activities to get rid of the above mentioned challenges.

3. PROPOSED FRAMEWORK

The proposed framework relates each development phase with its available artifacts, software metrics can be extracted from its artifacts and relates the test practices and QA best practices during the SDLC.
Section 3.1 introduces the framework architecture and its stages. Section 3.2 introduces the four classification dimensions of the framework activities. Section 3.3 introduces the project organizational structure and the team formation and their roles. Finally, Section 3.4 introduces the full description of the framework's phases their activities and their classification.

3.1 Framework Architecture

The framework architecture comprised into three distinct interleaved stages; the Pre-Development, Development and Post-Development stage. These three stages have a number of internal phases. Fig. 1 shows the framework components.

3.1.1 Software Pre-Development stage assure that the project commitments have been clearly defined considering a set of activities related to the project vision, resources required, the schedule and budget, project risk handling, business requirement and various project management plans through initiation and the planning phases.

3.1.2 Software development stage decomposed into six distinct, often overlapped / interleaved phases; requirement, design (High Level Design and Low Level Design) ,code and unit test, Pre-Testing, Test Execution (Higher Order Test) and Post-testing phases. For each phase in the development stage the possible created artifacts have been introduced to be able to monitor, review, predict and measure the progress of work done through these phases and assuring the quality of the produced release.

3.1.3 Software Post-Development stage concerned with managing the implementation of the produced release in the operational environment through deployment and operation phase.

3.2 Activities Classification's Dimensions

Software activities can be classified along the proposed framework into four dimensions; Management, Development, Review and Prediction and / or Estimation sector.

3.2.1 Management sector involves the development of project management plans (PMP) that provide a great control of the development activities and the introduction of managerial support actions that mainly prevent or minimize schedule and budget failures, continually changes in user requirements and identify and assess the risks resulting from the change one of the elements responsible for the software production through the various stages of the system.

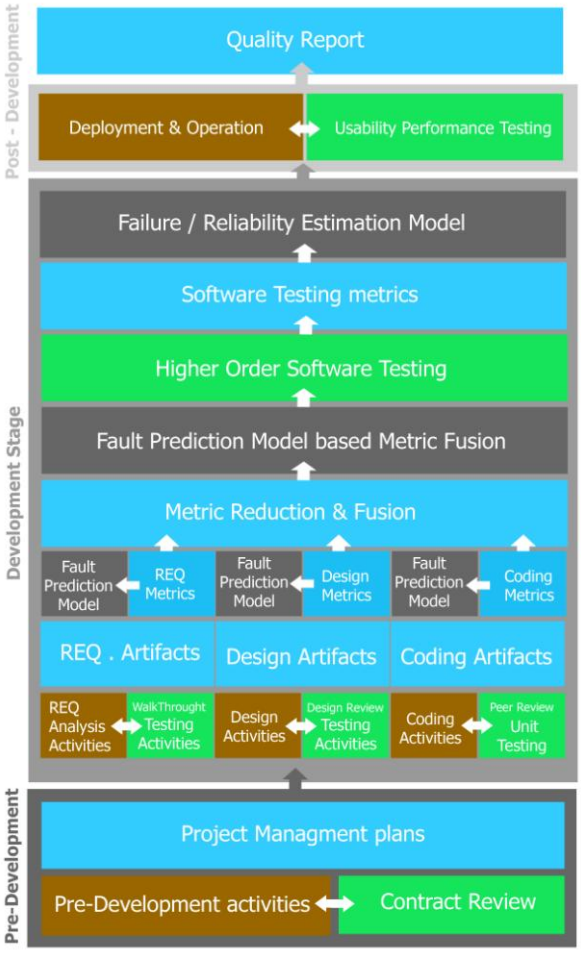

Fig 1: Framework Architecture.

3.2.2 Development sector includes a set of processes that concerned with developing the various deliverables of the release / product and managing their delivery to review sector for auditing and reviewing its quality and be ready for redeveloping or modifying its construction until it reaches the desired level of quality.

3.2.3 Review / Measurement Analysis sector includes a range of verification activities such as walkthrough, design review, peer review and inspection. It also locates the associated metrics for each software artifact that reflect the behavior of the software created through the development stage.

3.2.4 Prediction/ Estimation sector includes the construction of a range of models designed to empower the software testing process for improving the quality of the product and so by focusing on reducing the effort required to determine the presence or eliminate errors in product or assess the readiness of the product for use.

During the prediction part, fault prediction models that predict either fault-prone modules or fault content of the software modules based on collected metrics has been established. The failure estimation part of this sector attempts to estimate the current level of reliability of the software while in execution and determines the readiness of the system to release. Thus, the number of failures becomes the goal of estimation instead of faults. Fig.2 shows the classification of the fault prediction and estimation models and their implementation techniques.

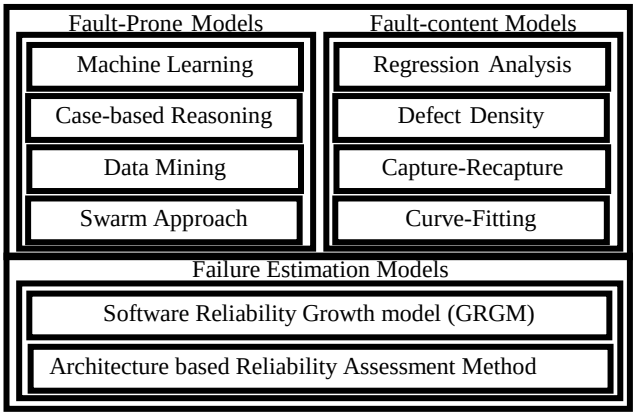

Fig 2: Fault prediction and failure estimation models and their implementation techniques.

3.3 Stakeholders Involvement

During each stage within the proposed framework, several stakeholders may be participated in the software project. A stakeholder is someone who has interest in the success of the project. Depending on their roles, different stakeholders may have different interests in the software project and involved in different software teams, such as; Planning, Dev, Testing, and System team. Table 1 illustrates the most typical project organizational structure and teams' formation.

Table 1. Project organizational structure and Teams' Formation

Stakeholder	Team Involvement				Interest		
	Planning Team (PT)	Dev. Team	Testing Team	System Team	Economic	Managerial	Technical
Customer	✓				▪		
User	✓	✓	✓	✓	▪		
PM	✓	✓	✓	✓	▪	▪	
HRM	✓					▪	
Master Developer	✓	✓	✓			▪	▪
S.W Developer		✓					▪
Master Tester	✓	✓	✓	✓		▪	▪
Tester			✓				▪
QA Lead	✓		✓	✓	▪	▪	
System Engineer	✓		✓	✓		▪	▪
Technical Support				✓			▪

3.4 Full description of the framework's phases

The following section explains in details the full description of the framework's activities. For each phase of the framework, the phase purpose followed by phase processes, their classification, and their associated artifacts have been established.

3.4.1 Initiation Phase

It concerned with the project vision, resource, and scope. Its main purpose is studying the feasibility of the project and initiating the project request. Table 2 illustrates the initiation phase's processes, their classification, and their associated artifacts.

3.4.2 Planning Phase

It concerned with the development of Project Management Plans (PMP) and establishing various planning activities. Table 3 illustrates the planning phase's processes, their classification, and their associated artifacts.

3.4.3 REQ. Analysis Phase

It concerned with producing the requirement documents both user and system requirements and its related testing artifacts. Once these artifacts have been documented, the review process done using walkthrough approach to produce the requirement metrics that used as basis for feeding the fault prediction model. Table 4 illustrates the REQ. analysis phase's processes, their classification, and their associated artifacts.

3.4.4 Design Phase

It concerned with the development of System Design Document (SDD) and its related testing artifacts such as unit and integration testing templates. Once the prototype has been created, the review process done using design review that updates static analysis report and produce design metrics used as basis for feeding the fault prediction model. Table 5 illustrates the design phase's processes, their classification, and their associated artifacts.

3.4.5 Coding Phase

During the coding phase the creation of the software product is underway and parts of the software product, such as classes, modules and subsystems are created. Unit testing represent a dynamic testing approach that test individual components of the system independently. Once peer review activities have been conducted, code metrics used as basis for feeding the fault prediction model are generated. Table 6 illustrates the coding phase's processes, their classification, and their associated artifacts.

3.4.6 Pre-Testing Phase

During the Pre-Test phase, Test case reduction has been conducted to select the most obvious test cases, and construct the software metrics repository that feed the fault prediction model that used mainly for managing the testing efforts. Table 7 illustrates the pre-testing phase's processes, their classification, and their associated artifacts.

3.4.7 Testing Execution Phase

After testing the individual components, a higher order test focus on the system as a whole and conducted as a series of testing issues that test the system from several perspectives including; Integration Test, Release Test, and Security Test. Table 8 illustrates the testing phase's processes, their classification, and their associated artifacts.

3.4.8 Post-Testing Phase

It concerned with assessing the readiness of the product to be released. Table 9 illustrates the post-testing phase's processes, their classification, and their associated artifacts.

3.4.9 Deployment and Operation Phase

It concerned with installing release into production environment, supporting its operation, reviewing its security and monitoring its performance. Table 10 illustrates the deployment and operation phase's processes, their classification, and their associated artifacts.

Table 2. Initiation Phase's processes, their classification, and associated artifacts

Initiation Processes	Classification				Artifact	Roles	
	Management	Development	Review	Prediction/ Estimation		Responsible	Beneficiary
Initiate Business Case & Scope	▪				PSD	PM, Customer	PT
Establish Team Structure	▪				organizational chart	PM, HRM	PT
Identify Preliminary Risks	▪				Risk Assessment	PM	PT
Establish project Proposal	▪				Project Proposal	PM, Customer	PT
Establish Project Charter	▪				Project Charter	PT	PT
Contract Review			▪		Feasibility Report	PM, Customer	PT
Phase Status Review			▪		Status Report	PM	PT

Table 3. Planning Phase's processes, their classification, and its artifacts

Planning Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Establish WBS	▪				WBS	PM	PT
Define Detailed Roles	▪				RAM	PM, HRM	PT
Estimate Efforts	▪				Staff Plans Resource Plans	PM, HRM	PT
Scheduling	▪				Schedule Plans	PM	PT
Develop PMP	▪				PMP	PT	All Stakeholders
Phase Status Review			▪		Status Report	PM	PT

Table 4. REQ. analysis Phase's processes, their classification, and associated artifacts

REQ. Processes	Classification				Artifact	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Establish Risk Management	▪				Risk Assessment	PM	
Establish Contingency Plans	▪				Update PMP		PT
Develop Concept Of Operations		▪			URD, AUT	Dev Team	All Stakeholders
REQ Analysis, Prioritization		▪			SRD, RTM	Dev Team	Dev Team
MTP formulation		▪			MTP	Master Tester	Testing Team
REQ Review			▪		REQ Metrics, SAR	QA Lead, Master Developer	Dev, Testing Team
Phase Status Review			▪		Status Report	PM	PT , Dev Team
Prediction based REQ. Metrics				▪	TAR	QA Lead, Master Tester	Testing Team

Table 5. Design Phase's processes, their classification, and associated artifacts

Design Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Handle REQ Change	▪				Update RTM	PM	PT
Establish Contingency Plans	▪				Update PMP		
Establish Risk Management	▪				Risk Assessment		
Develop Security Plans	▪				Security Plans	Master Tester	Testing, System Team
Develop Integration Plans	▪				Integration Plans	System Engineer	Testing, System Team
Develop Architecture Design		▪			SDD	Architect, Master Dev, DB Admin	Dev, Testing, System Team
Interface Design		▪			Interface Specification		
Database Design		▪			DST		
Develop Prototype		▪			Prototype		
Test Case Design		▪			UTC, ITS	Master Tester	Dev, Testing Team
Design Review			▪		Design Metrics, SAR	QA Lead, Master Developer	Dev, Testing Team
Phase Status Review			▪		Status Report	PM	Dev Team, PT
Prediction based Design Metrics				▪	TAR	QA Lead, Master Tester	Testing Team

Table 6. Coding Phase's processes, their classification, and associated artifacts

Coding Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Handle REQ Change	▪				Update RTM	PM	PT
Establish Contingency Plans	▪				Update PMP		
Establish Risk Management	▪				Risk Assessment		
Establish Dev Environment		▪			System Dev. Doc	Master Developer	Dev Team
Build Release units		▪			Release unit	Developer	Dev , Testing Team
Conduct Unit Testing		▪			Defect Log		
Peer Review			▪		SAR, Coding Metrics	QA Lead, Master Developer	Dev , Testing Team
Phase Status Review			▪		Status Report	PM	Dev Team , PT
Prediction based Coding Metrics				▪	TAR	QA Lead, , Master Tester	Testing Team

Table 7. Pre-Testing Phase's processes, their classification, and associated artifacts

Pre-Testing Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Establish Risk Management	▪				Risk Assessment	PM	PT
Establish Contingency Plans	▪				Update PMP		
Test Case Reduction		▪			Test Case Repository	Master Tester	Testing Team
Development's Metric Collection & Analysis		▪			Metrics Repository	QA Lead	Testing Team
Phase Status Review			▪		Status Report	PM	PT, Testing Team
Prediction based Reduced Metrics				▪	TAR	QA Lead, Master Tester	Testing Team

Table 8. Testing Phase's processes, their classification, and associated artifacts

Testing Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Establish Risk Management	▪				Risk Assessment	PM	PT
Establish Contingency Plans	▪				Update PMP		
Establish Testing Environment	▪				STD	Master Tester	Testing Team
Conduct integration Test		▪			Defect Log TAR	Tester	Dev, Testing Team
Conduct Release Testing		▪					
Conduct Security Testing		▪					
Phase Status Review			▪		Status Report	PM	PT, Dev, Testing, System Team

Table 9. Post-Testing Phase's processes, their classification, and associated artifacts

Post-Testing Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Establish Risk Management	▪				Risk Assessment	PM	PT
Establish Contingency Plans	▪				Update PMP		
Plan for Deployment	▪				Training Plans, User Manual	System Engineer	Customer, Dev., Testing Team
Conduct Acceptance Test		▪			TAA	Master Tester, User	System Testing
Complete Release Documentation		▪			Release Documentation	Testing Team	All Stakeholders
Phase Status Review			▪		Status Report	PM	PT, Testing Team
Conduct Reliability Estimation				▪	Readiness Report	QA Lead, Master Tester	Customer, Dev., Testing Team

Table 10. Deployment and Operation Phase's processes, their classification, and associated artifacts

Deployment and Operation Processes	Classification				Artifacts	Roles	
	Management	Development	Review	Prediction / Estimation		Responsible	Beneficiary
Establish Risk Management	▪				Risk Assessment	PM	PT
Establish Contingency Plans	▪				Update PMP		
Deploy Release in Operational Environment		▪			Complete Release	Technical Support	System Testing
Conduct Training Plans		▪			RTR USRR	User, System Team	All Stakeholders
Support Release Operation		▪					
Review Release Security			▪				
Review Release Performance			▪				

4. CONCLUSION

The framework introduces a completely integrated view of the development activities and their related artifacts treating the testing and QA practices as integral parts of the development process.

The framework promotes the efficiency of the stakeholders through the redistribution of their roles in the different teams. It introduces a new role relating the managerial and technical interest at development and testing levels to find a linkage between members of different teams in the project without the overlap in their roles, So that teams work all harmoniously tight proceeding according to system administrative work tightly in line with the plan developed for the project as a whole.

We aim to touch in our future studies the construction of a software quality improvement model based on the measurement analysis at various levels of the proposed activities classification's dimensions.

5. REFERENCES

[1] Yu BengLeau, WooiKhong Loo Wai Yip Tham, and Soo Fun Tan, "Software Development Life Cycle AGILE vs. Traditional Approaches", International Conference on Information and Network Technology (ICINT), 2012, pp. 162-167.

[2] Shikhamaheshwari, Dinesh Ch. Jain, "A Comparative Analysis of Different types of Models in Software Development Life Cycle", International Journal of Advanced Research in Computer Science and Software Engineering, Vol.2, No.5, 2012, pp.285-290.

[3] Kirti Nagpal,Raman Chawla, "Improvement of Software Development Process: A new SDLC Model", International Journal of Latest Research in Science and Technology, Vol. 1, No.3, 2012, pp.217-224.

[4] T.Rajani Devi, "Importance of Testing in Software Development Life Cycle", International Journal of Scientific & Engineering Research, Vol.3, No.5, 2012, pp.1-5.

[5] Jamie S. Gordon, Robert F. Roggio, " A Comparison of Software Testing Using the Object-Oriented Paradigm and Traditional Testing", Proceedings of the Conference for Information Systems Applied Research, San Antonio, Texas, USA , Vol. 6, No.28, 2013 .

[6] Rashdeep Kaur, Sunil Gulati," A framework for Analyzing Software Quality using Hierarchical Clustering", International Journal on Computer Science and Engineering (IJCSE), Vol.3, No.2, 2011, pp.854-860.

[7] T.M.H koshgoftaar, P.Rebours and N.seliya," Software Quality analysis by combining multiple projects and learners", software Quality Journal, Vol.17, 2009, pp.25-49.

[8] S.Shafi,S.M.Hassan,A.Arshaq,M.J.Khan,S.Shamail, "Software Quality Prediction Techniques: A comparative Analysis ", 4th International Conference on Emerging Technologies (ICET),, 2008, pp.242-246.

[9] Heena Kapila, Satwinder singh, "Analysis of CK Metrics to predict Software Fault-Proneness using Bayesian Inference", International Journal of Computer Applications (IJCA), Vol. 74, No. 2, 2013, pp. 1-4.

[10] Milad Hanna, Nahla El-haggar and Mostafa Sami. Article: Reducing Testing Effort using Automation", International Journal of Computer Applications, Vol. 81, No. 8, 2013, PP.16-21.

[11] Hilda B. Klasky, "A study of Software Mertics", M.S. thesis, Department of Electrical and Computer Engineering, Rutgers University, New Brunswick, New Jersey, 2003.

[12] Saddam H. Ahmed, Taysir Hassan A. Soliman , and Adel A. Sewisy, " A Hybrid Metrics Suite for Evaluating Object-Oriented Design", International Journal of Software Engineering, Vol. 6, No. 1, 2013, pp. 65-82.

[13] D. Jeyamala, S. Balamurugan, A. Jalila, K. Sabari Nathan, " Fault-prone Components Identification for Real Time Complex Systems Based on Criticality Analysis", International Journal of Computer Science and Informatics, Vol. 3, No. 2, 2013, pp. 17-23.

[14] Gurvinder Singh, Baljit Singh Saini , Neeraj Mohan, " A Systematic Literature Review on Software Fault Prediction based on Qualitative and Quantitative Factors", International Journal on Advanced Computer Theory and Engineering (IJACTE), Vol.2, No. 2, 2013, pp. 47-50.

[15] Shaik Nafeez Umar et al, " Software Testing Defect Prediction Model -A Practical Approach", International Journal of Research in Engineering and Technology (IJRET), Vol.2, No.5, 2013, pp. 741-745.

[16] Menka Gupta, PratimaGautam, " A Novel Approach for Identifying Software Fault Prediction in mining", International Journal of Technological Exploratoration and Learning (IJTEL), Vol.2, No.6, 2013, pp. 341-343.

[17] Nirvikar Katiyar, Raghuraj Singh, " Prediction of Number of Faults And Time To Remove Errors", International Journal of Computational Engineering Research, Vol.3, No.4, 2013, pp. 57-65.

[18] S.Kim, T.Zimmermann, E.J.W.Jr., and A.Zeller, " Predicting faults from cached history",29th International Conference on Software Engineering (ICSE), 2013, pp.489-498.

[19] T.J.Ostrand, E.J.Weyuker, and R.M.Bell, " Predicting the location and number of faults in large software system", IEEE Transactions on Software Engineering, Vol.31, No.4, 2005, pp. 340-355.

[20] Anshu Bansal, Sudhir Pundir, " A Review On Approaches and Models Proposed for Software Reliability Testing", International Journal of Computer & Communication Technology, Vol.4, No.2, 2013, pp. 7-9.

[21] J Prabhu and N Malmurugan, "Article: A Model for GUI Automated Testing Framework in Software System", International Journal of Computer Applications, Vol.64, NO.15, 2013,PP. 16-20.

[22] Rakesh Roshan, RabinsPorwal, Chandra Mani Sharma, "Review of Search Techniques in Software Testing ", international Journal of Computer Applications, Vol.51, No.6, 2012, pp.42-45.

[23] Dinesh Kumar Saini., "Article: Software Testing for Embedded Systems", International Journal of Computer Applications, Vol.43,No.17,2012,PP.1-6 .

[24] Anuja Jain M, Swarnalatha P, Muhammad Rukunuddin Ghalib and S Prabu. Article: Web-Based Automation Testing Framework", International Journal of Computer Applications, Vol. 45, No. 16, 2012, PP.1-5.

[25] Anudeep Ediga, " High Accuracy Prediction Framework for Predicting Defect Prone Software Components", International Journal of Information and Computation Technology, Vol.3,No.10, 2013, pp.985-992.

Table 11. Appendix A – Abbreviations

AUT	Acceptance User Test
BA	Business Analyst
DST	Data Storage Design
GUI	Graphical User Interface
HRM	Human Resource Manager
ITS	Integration Test Script
MTP	Master Test Plan
PM	Project Manager
PMP	Project Management Plan
PSD	Project Scope Document
PT	Planning Team
QA	Quality Assurance
RAM	Responsibilities Matrix
RTM	Requirement Traceability Matrix
RTR	Release Troubles Report
SAR	Static Analysis Report
SDD	System Design Document
SRD	System Requirement Document
STD	System Testing Document
TAA	Test Analysis Approval
TAR	Test Analysis Report
URD	User Requirement Document
USRR	User Satisfaction Review Report
UTC	User Test Case
WBS	Work Breakdown Structure