I.J. Modern Education and Computer Science, 2016, 7, 53-60
Published Online July 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2016.07.06

An Analysis of Performance Testing in Distributed Software Applications

Muhammad Fraz Malik
Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad, Pakistan
Email: urd.logic@gmail.com

M. N. A. Khan
Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad, Pakistan
Email: mnak2010@gmail.com

***Abstract*—**Testing is a crucial step in designing and implementing software in the distributed environment. Testing in the distributed applications is not only difficult, but also a costly method. This Research briefly discusses the performance testing in the distributed software environment along with different other testing techniques proposed in the literature that correspond to distributed applications. Additionally, we discuss the key testing challenges faced during the whole process of testing the distributed applications. Much of the focus of this paper is on intelligent agent-based testing. Agent based testing is provide better coordination mechanism between multiple testers and exert more controllability and observablility on fault detection. In this study we have critically analyzed testing methodologies being practiced in the distributed environment. We have studied the merits and limitations of these methodologies proposed in the contemporary literature and have identified the possible improvements in those methodologies to make them more robust.

***Index Terms*—**Performance Testing, Distributed Applications Testing, Agent-based Testing, Random Testing.

I. Introduction

Software testing is one of the most difficult tasks to assure the quality of software and plays a vital role in the SDLC process to ensure that it adheres to the client or customer needs. When it comes to the distributed applications environment, software testing is considered as backbone for applications. Testing the distributed software application is much more difficult as compared to testing standalone software applications, because the distributed system behaviors are dynamically changed with respect to time and platforms. There are several key challenges linked to testing the distributed applications; e.g., the same test run executed frequently on the same scenario with same input, may generate different outputs. This happens due to the non-linear behavior of the distributed systems i.e., event timing can also affect the end results. The functional and non-functional requirements play key role in web applications testing.

Some sort of software package assessment method describes a few phases in different assessment situations.

The approaches and the testing methods used to functional requirement for traditional software as well as used for web application. The non-functional testing includes performance testing, load testing, stress testing, compatibility testing, usability testing, accessibility testing and security testing etc. performance testing is much important for the insurance of reliability and efficiency for web application. During implementation, performance related issues can be resolved is an efficient way to beat the market challenges as well as client requirements. Performance testing is used to test the performance of the application by integrated the software systems, and it's a more effective testing technique as compared to the functional testing. There are three main qualities for applying additional reliability, accurate and efficiency of distributed software testing applications. Scalability and response time are key attributes for performance characteristics in distributed software applications. This study identifies some common issues that testers face while managing distributed software system.

Software testing is a process to assess software behavior according to the desired user specifications and to ensure quality of the product. The abnormal software behavior is normally termed as a bug which could be an error, fault, flaw or failure of a computer program that causes it to produce unexpected result. The sole purpose of testing is to find software failures so that the bugs can be discovered and corrected. Software systems differ in the manners in which they fail. Mostly, physical systems fail in a pre-defined set of ways, but a software system could fail in many different strange ways making it difficult for the testers to discover all those possibilities. Therefore, new testing standards and novel testing techniques are continuously emerging.

Testing has a prime role in software development life cycle (SDLC) and the overall software system architecture as it is vital to assure the software quality before its deployment and during its maintenance. There are several testing methods which are organized into the following basic categories:

- Static [Reviews, walkthroughs and inspections]
- Dynamic [Execution of code with test cases]
- White-Box [Testing internal structure or working of a program]
- Black-Box[Testing only working of the program]
- Visual [Testing the Graphical User Interface]
- Grey-Box [Somewhat in the middle of White-Box and Black-Box]

Testing can be performed at several levels such as:

- Unit [Testing individual unit of source code]
- Integration [Software modules are combined and tested as a group]
- Component Interface [Testing data handling across various components]
- System [Testing complete system against specifications]
- Acceptance [User of the system will perform acceptance testing]

A number of application programs have been developed to automate the testing process to reduce human error. CODESONAR is one of the tools to automatically perform static analysis of software behaviour.

Requirements are actually customer's statements of scope. In requirements finalization process, the stakeholders play an import role. A stakeholder can be defined as anyone who is directly or indirectly affected by the system being developed or deployed. Stakeholders are broadly categorized into two major classes — user and customer. User ordinarily uses the system and customer refers to those persons who have requested for the development of the system and are responsible for approving it. There may be a number of people who participate in the development of a system like business analysts, designers, coders, testers, project managers, deployment managers, use case designers, graphic designers etc. and are customarily considered as stakeholders. Requirements engineering (RE) is a group of activities to elicit, analyze, specify, verify, validate and manage requirements.

To ascertain their quality requirements practices product managers from Swedish companies were interviewed. The study found that usability and performance requirements were being considered as the most important. The study also identified that a major challenge with quality requirements was to make it well specified and quantified so that to make it testable. Survey in Canada showed that inadequate testing; training and lack of formal criteria for testing increase the risk of releasing defects into production and release versions of software products. The ability to detect tendencies that lead to reduced quality and to identify the root causes of reductions in product quality suffer from the lack of exhaustive testing of the product. There is a need for improvements for quality assurance. Organizations that offer training on quality assurance and testing procedure to their developers improve their ability to validate quality and correspondingly improve the likelihood of pinpointing and resolving process and product defects. Cost and lack of expertise are two major barriers for adaptation of testing methodology and tools. Research in Brazil conducted with objective to identify respondents' perception about the relationship between IT Governance models and quality instruments adoption. The results obtained through the survey provided an overview of the impact of IT-related problems in the organizations, the degree of knowledge and importance as well as adoption of different quality instruments [14]. A survey on quality models being practiced in Germany focused on several domains of software development such as standard software being used, custom development methodologies, embedded systems and company size. Exploratory survey on software practices of software firms in five ASEAN countries show that software industry was falling short in the four areas: use of automated estimation tools (Software Project Planning Practice), use of requirements traceability matrices (Software Project Tracking and Oversight practice), use of quantitative quality metrics (Software Quality Assurance practice), and the use of change control boards (Software Configuration Management practice). Organizations that were strong in software quality assurance practice had in common the best practices of quality orientation, independent testing teams and Software Engineering Process Group (SEPGs), peer reviews, and quality management systems.

The remaining paper distribution is as follows: Section I provides an introduction highlighting the importance of distributed software applications. Section II is about research objective. Section III is about literature review, in this section we discuss the topic related research paper which we reviewed. Section IV provides critical analysis, regarding technical strengths and weaknesses of the surveyed literature. Section V describes conclusion and future work.

II. Research Objectives

The objective of this research is to discover strengths and weaknesses of the existing software testing techniques particularly the ones that support distributed environment. Based on the merits and demerits of the existing techniques, specific research gaps would be identified which could help find possible future directions in this area. In addition, cloud based testing tools and services would also be studied. The study will serve as a survey report highlighting the efficacy, usefulness and strengths of different software testing techniques particularly in the distributed environment. In this study, I will conduct a systematic literature review of the software testing techniques, their areas of applications, the approaches used in the formation test oracle and test cases and SWOT analysis of the existing testing techniques. The scope of the study is limited to review and critically analyze the existing software testing techniques. The study would mainly be a survey report about the merits, demerits of the software testing techniques. In this study, research method will be kind of a meta-analysis targeting critical review of the latest papers published in the domain of software testing. For this purpose, research papers published during the last five years in the area of Software Testing in Distributed Environment will be studied, reviewed and critically analyzed. In addition, the strengths and limitations of the proposed techniques in the area of Software Testing will be determined. The output of this non-empirical research approach is envisaged to locate research gaps in the thrust for finding potential future works in this domain.

This paper focuses on the following objectives:

- Discuss the overview of distributed software Application and testing.
- Describe the main role of performance testing and identifying some common issues in distributed software application.
- And identify the common frameworks. Discuss their merits and de merits and along with future work.

A. Distributed Software Application

In distributed software system the different components located of different locations can communicate and coordinate their actions by passing the messages. All processor have equal rights to access shared memory between the distributed computing for exchanging their information between the multiple processors. Distributed software application discussed below. Examples of distributed applications are:

- Wide Area Network Application(WAN)
- Wireless Sensor Networks
- World Wide Web(WWW)
- Distributed Database Management Systems
- Banking system and Airline reservation system are key example of Distributed information processing

B. Distributed Software Testing

Software testing is required to check that the completed application is according to the user requirement. Developer of software can deliver software when meet the user expectation and needs. By using the software testing, we identifying the correctness, completeness and qualify of software on the Software Development Life Cycle (SDLC). Software testing process ensured that software stands well against the define requirements.

Basically testing is process which is used to investigate and perform on the behalf of stockholder. The earlier testing was done after code has been completed but now a day's testing is begin in design the test cases. These test cases fall on three testing categories. Gray box testing, Black box testing and White box testing.

III. Related Work

This section is described about different testing methodologies in different distributed software application. Based on the literature reviewed in this study, we categorize the distributed testing approaches into the following set of domains as described in the subsequent sections.

Nowadays, many organizations are investing huge amount of money to improve the quality of their software and software development process; and are using different quality assurance processes, methodologies and practices to improve the quality of their software. Code reviews, walkthroughs, standards and procedures are consistently checked through performance monitoring, product evaluation, inspection, validation, verification, audits and allied testing activities.

A. Distributed Testing with Agile Methodology

Collins et al [1] combine the characteristics of two methods: DSD (Distributed Software Development) and agile software development methods. According to the authors, the old works did not cover the scenario where the testing activities are distributed among different teams which are geographically separated. Based on that study, the authors advocate which the software development along with their own testing collaboration will be difficult if your team members usually are geographically separated. And also to similar other factors, e.g. work time, cultural differences, communication gaps and technical incompatibilities etc. these all factors may impact the success of the software project. The main contribution of this research is to highlight the challenges being faced in the distributed testing environment and worked out a methodology to conduct distributed application testing using agile software development technique. Testing is generally conducted late in the development process. Testing mainly aims at detecting the defects. The authors highlight the importance of preventing the defects. In this research, the authors used agile development method in distributed software testing.

B. Testing Web-Based Application

Di Lucca et al. [2] Performed analysis of different testing method for web applications with respect to functional and nonfunctional requirements. For web application, the research highlights that functionality depends on following aspects: testing levels, testing strategies, test cases, testing models and test processes.

C. Adaptive and Random Partition Software Testing

Adaptive testing is a feedback-based software testing strategy that has been more effective than Random Testing (RT) and Partition Testing (PT). A major concern in the application of AT is its complexity and computational cost for test case selection. Lv et al [8].

Propose hybrid approach that uses AT and Random Partition Testing (RPT) is an alternating manner. The motivation for this approach is that both strategies are employed such that the underlying computational complexity of Adaptive Testing is reduced by introducing Random Partition Testing (RPT) into the testing process without affecting the defect detection effectiveness. A case study with seven real-life subject programs is presented in the study.

D. A Temporal Agent Based Approach for Testing Distributed Systems

Coordination and communication are more complex features of the distributed testing components. For such general reactions for errors, Time outs, observability, locks, controllability and synchronization problem are build. Azzouzi et al. [13] focuses on the temporal properties that specify the time required for exchanging messages between the various components of the distributed test applications. The study introduces new architecture to avoid the synchronization problem between different testers. The aim of the study is to propose better coordination mechanism between multiple testers and exert more controllability and observablility on fault detection. Another objective is to check timing constraints in distributed testing correctly in such a way that all testers and clocks should be synchronized. This development with the distributed testing framework is a difficult process where the testing system must not only check if the output events have been observed, and also the time whenever most of these events have been happened. The project presented within this research extends result from testing in distributed technique to deal with testing an implementation under test with some testing constraint.

When ensure the reliability and efficiency of Web Services, performance testing on web services becomes very important. But the measurement of performance on Web Services very difficult. Hao et al. [12] introduce the method of performance testing, and proposed a framework of agent-based performance testing on web services, which include the some features e.g. “Test Flow Generator “which is used to generates test cases and test flows by WSDL (Web Service Description Language) files and user demand. “Scenario Creator” create scenario by choose the required schedule and selecting the required agent test flows and the quantity. “Test Manager “It gets real time data from agent and monitors the rest result by displaying into in chart. “Load Generator Agent “is used to balance the pressure of Test manager and increase the load. And the last “Test Analyzer” this module is used to analysis test results. The implementation of kernel Modules introduced specifically in this framework. To realize the load allocation to distributed load Generation agents from test manager, and provide a queue based allocation strategy.

Provide better reliability and efficiency of web services. Improve the performance of web applications with the help of agent based testing. The author design a framework which is used to measure the performance testing of distributed multi-agent on web service, This framework is not only provide s a model for the automated performance testing on web services but also a general framework for testing of other protocols. This framework has two interfaces, one is user friendly interface, test system user (TSU), and the other is system under test (SUT) interactive with tested system. The following modules are included, test flow generator, Scenario creater, test manager, load generation agents.

The key reason of conducting this research because the performance testing of web services are resolved the following issues. Operations of multiple concurrent users are simulated by multiple SOAP message. Concurrent users should be dispatched and managed centralized by the master. Concurrent users should send test commands to tested system in distributed physical nodes. Dynamicity and adaptiveness that limit the ability of the tester to determine the web services that are invoked during the execution of a workflow.

The Strength of research is the framework Analyze automatically the log and generate test report. And combine the characteristics of functional testing and performance testing. Performance testing focus the efficiency, reliability and accuracy of the framework.

The cost of using a service (for services with access quotas or per-use basis). This research proposed a framework of performance testing on web services and implements the kernel modules including scenario creator, test manager and load generated agent. This is used to maintain the consistency of test environment and real environment and balance load generated by multiple concurrent threads. The framework will be refined with more formal definition of Agent based testing, rules, test plan, and other testing protocols. Moreover, the mechanism of rule-based test planning and dynamic test agents will be further explored.

E. Dynamic Testing Tool for Agent-based Systems

Eassa et al. [10] introduce a dynamic testing tool that use a temporal logic assertion language for detecting run time errors in agents and agent based systems. This tool *evaluates* agent behavior and detect errors related to agent behavioral constraints, errors related to agents ‘interactions, errors related to user requirements and security of agent based web applications. The proposed technique is based on the grammar, the syntax and the semantic of the temporal logic assertion language. In this research the dynamic testing tool has been built and tested for ascertaining its effectiveness against its use as a dynamic testing tool.

F. Agent Based Software Testing with Role Oriented Method

Agent oriented software engineering methodologies provide us a platform to develop agents based systems. These methodologies mainly focus on development rather than the testing. It is not possible to map all agent properties e.g. autonomy, reactivity etc. to object oriented constructs. Therefore, a proper testing technique for agent-based software solutions is needed. Sivakumar et al

[6]. Propose an effective and specialized testing technique for agent-based systems. The proposed technique focuses the main attribute of an agent which is role. It follows a v-based model which starts from requirements and ends at role-based acceptance testing. The proposed approach provides better solution for industrial, commercial, medical, networking and educational applications related problems. This research is based on an effective agent-oriented framework which provides high quality software development process and products.

Loffler et al. [14] discussed a method of agile which is scrum. This method has much customer participation. It gives ease of techniques that are used in the analysis of requirements and testing. This model is consists of different useful techniques which are important in maintaining the record of users requirements. These techniques are in the nature models that are already tested. This model can be used by both testers and developers. It is easy to adopt and language used in it is simple. Test scripts derived from this model for fitNesse / selenium. Scrum focal point is implementation. New and users preferred features are included in previous design by using iterative steps. They are known as sprints. When these sprints are included, they caused deprived and deficient requirements. As the consequences user feel trouble to using software. At some stage in sprints the next problem is continuous addition of new functions and their implementation. These problems are like latest functions are not harmful for existing ones. References [15-48] reviewed different techniques in different domains and reported their critical evaluations.

IV. Crtical Analysis

In this section we critically analyze our research topic completely: with respect to their key strength, as well as discuss their Limitations and the future improvements. This is basically comparative study related to literature review.

Table 1. Summary of testing in distributed software application.

Ref #	Research topic	Strengths	Limitation	Suggested Improvements
[1]	Distributed testing with Agile methodology.	The proposed solution reduces the impact of communication issues on the software project.	The communications and coordination are key factors of distributed testing environment and absence of these two factors would lead to failure of the proposed approach.	The probable improvement could be to calculate testing efforts required to perform such type of testing to evaluate its true worth.
[2]	Testing in Web-based applications	By composing different hardware for heterogeneous execution environments which includes the network connections, operating system, web server and web browser.	The main problem to evaluate the performance of web application is that users always expect that services response on time and users usually don't waits for long requests.	The possible improvement could be discovering effective approach web services testing due to the introduction of 'Agile' methods in web application testing process
[3]	Testing is used to maintain scientific software	This study is conducted in real environment and provides concrete statistics in terms of efficiency, effectiveness and cost estimation (manpower, time, etc).	This scientific software used Regression testing. This testing is less effective when new features are introducing to the software. The implication of the study is limited to the scientific software only.	Regression tests are normally performed at higher level (at input and output level) and this can be automated.
[4]	Distributed in vivo testing of software applications.	When the couple of applications worked together may chance to increase the probability that an instance will be finding the erroneous states easily.	A major limitation of in vivo testing is the high performance cost it insure as the unit tests are executed in parallel with the application.	To overcome huge cost of replicating the process, one way to limit the numbers of tests which are run together by assigning portions of test suite to different members of the community.
[5]	Distributed testing in Database software applications.	In this research the authors used sanitize information in centric objects to make it difficult for attackers to infer sensitive information. The PISTIS technique runs faster and thus can be applicable in real world testing scenarios.	Many data anonymization algorithm seriously degrade test coverage of database-centric applications.	The possible improvement for implementation of PISTIS is to reduce the analysis time and other bottlenecks while using data mining approach which are computationally intensive.
[6]	Agent Oriented Software Testing–Role Oriented approach	The proposed approach provides The determination of efficiency in different testing techniques and this efficiency is used to generate the quality of software.	This approach does not provide a proper testing framework for testing the agent based software.	This approach enhanced with agent based system was tested using with object-oriented testing method. And the result provide with tabulated form.
[7]	Incremental test generation for software product lines	Introduce new approach for classical testing domain in which tailoring the practices with respect to the requirements of the software product lines.	For the lack of space considerations the lager scope of experiments results are not presented.	The possible improvement of incremental approach for software products line to hold promises in lager software testing context e.g., for refining tests for regression testing.
[8]	Adaptive and Random	While choosing the test cases, the RT and RPT are efficient testing strategies	The disadvantage of RT is its defect detection effectiveness as	The future improvement of this methodology could include different

	Partition Software Testing.	with respect to the computational overhead and cost.	much effort has been invested to improve the effectiveness of RT and RPT	heuristic methods, a theoretical analysis on balancing defect detection and testing efficiency, and optimal parameter settings for different subject programs under testing.
[9]	Code coverage of adaptive random testing	The Adoptive random testing provides a higher effectiveness of software application with respect to reliability, and a higher confidence. And reliability of the software under test even when no failure is detected.	In ART and RT the selection process depends on tester knowledge and experience for different test cases. It means that a tester needs to identify appropriate categories and choices for the program under test.	There are various ART algorithms proposed in the literature which evenly distribute random test cases in different ways. A possible future work could be the code coverage measuring and comparing by different art Algorithms.
[10]	Dynamic Testing Tool in Agent-based Systems	The proposed technique used mobile agents for software testing tasks. And provides the abstract model that is used to generate abstract test cases. Those test cases are then concretized and addressed to the system under test. Another key reason is to a test automation framework for multi-agent system that enables developers to build and run test scenarios.	This testing is naturally bound to a limited testing time and to provide a small number of execution conditions.	The proposed technique can be enhanced to account for detecting the interaction errors between mobile agents and all communication errors between agents and environments. Another future work could be to enhance the dynamic testing tool to detect other security vulnerabilities such as of “attacker access to the flaw” in agent-based web applications
[11]	By using different methods for debugging the multi-agent system software tests.	This paper proposes the use of relational database (a very powerful tool for querying data) server as a central storage mechanism.	The infrastructure of MAS requires costly maintenance. Every change in code of MAS infrastructure would be very costly because it needs to be replicated.	Reducing the faults automatically in the MAS for collaboration, order and abstract graphs are possible future work Which involves the to define the expected results into a specific languages.
[12]	Framework for testing the performance on distributed agent based web services.	The proposed approach analyzes the log automatically and then generates test report. It also combines the features of performance testing as well as functional testing and focus on the reliability and accuracy of the system.	Dynamicity and addictiveness limit is based on the ability of the tester to decide the web services invoke during the execution of the workflow.	The framework also enhanced by refining the formal definition of agent based testing’s rules, test plan and other testing protocols. The mechanism of dynamic test agents and rule based test planning can be further explored.
[13]	A temporal agent based approach for testing open distributed systems	In timing constraints prospective, the distributed testing process not only checks different output events, but also records the timestamps when these events have occurred. The correctness of testing in distributed systems depends not only on the logical result of a computation, but also keeping tracks of the time when the result was delivered.	Sometime serious coordination issues occur among different testers when they communicate. Many problems influencing fault detection during the conformance testing process arises if there is no coordination between distributed testers. This problem is known as controllability and observablility fault detection.	possible future extension to this work could be adding a timing constraint

V. Conclusion And Future Work

This study focuses on different testing methodologies in distributed software applications. The contributions of the study include: identifying the possible techniques for testing distributed software applications followed by identifying the challenges faced during the testing in software application. In addition to the identification of key challenges related to application testing in the distributed environment, we also provide the possible solution to overcome these challenges. As a future dimension to this study, we intend to focus on agent-based testing in the distributed environment. Agent based testing is provide better coordination mechanism between multiple testers and exert more controllability and observablility on fault detection it might automatically create and by organizing test agents which they decompose the task into subtasks for testing of entire distributed application; and this servers as an impetus to conduct IS-2 research on this topic. This study also finds that better coordination and communication among different test agents can reduce the cost and several other issues such as scalability and reliability in the distributed testing environment.

References

[1] E. Collins, G. Macedo, N. Maia., and A. Dias-Neto,” A Industrial Experience on the Application of Distributed Testing in an Agile Software Development Environment”, In *Global Software Engineering (ICGSE) on IEEE Seventh International Conference, pp-190-194,2012.*

[2] G.A Di Lucca, and A, R Fasolino,” Testing Web-based applications: The state of the art and future trends”on *Information and Software Technology, vol 48, pp-1172-1186, 2006..*

[3] T. Clune, M. Rilee, and D. Rouson, “Testing as an essential process for developing and maintaining scientific software”.on In *The 2nd Workshop on* Sustainable Software for Science: Practices and Experiences, 2014.

[4] M. Chu, C. Murphy, and G. Kaiser, “Distributed in vivo testing of software applications. In *Software Testing Verification, and Validation”, on 1st International Conference on,* pp. 509-512, 2008

[5] B. Li, M. Grechanik, and D. Poshyvanyk, “Sanitizing and minimizing databases for software application test outsourcing. In *Software Testing, Verification and*

Validation (ICST)" on IEEE Seventh International Conference on, pp. 233-242,2014.

[6] N. Sivakumar, and k. Vivekanandan, "Agent Oriented Software Testing–Role Oriented approach" on *International Journal of Advanced Computer Science and Applications, vol 3, 2012*

[7] E. Uzuncoava, S.khurshid,, and D. Batory, "Incremental test generation for software product lines" on *Software Engineering, IEEE Transactions, vol 36, pp.* 309-322,2010. [8] Lv, J., Hu, H., Cai, K. Y., & Chen, T. Y. (2014). Adaptive and Random Partition Software Testing.

[8] J. Lv, H. Hu, K. Y. Cai, and T. Y Chen," Adaptive and Random Partition Software Testing",2014.

[9] T. Chen, F., Kuo, H., Liu and E. Wong," Code coverage of adaptive random testing", *on IEEE Transactions on Reliability, vol 62, pp.* 226-237.2013.

[10] F.E, Eassa., L.J Osterweil, M. A, Fadel, S. Sandokji and A Ezz," DTTAS: A Dynamic Testing Tool for Agent-based Systems"on *Pensee Journal, vol 76. 2014*

[11] E. Serrano., A. Munoz,. And J. Botia." An approach to debug interactions in multi-agent system software tests." On *Information Sciences,vol 205, pp.* 38-57,2012

[12] D. Hao, Y., Chen., F. Tang, and F. Qi. "Distributed agent-based performance testing framework on Web Services" In *Software Engineering and Service Sciences (ICSESS)* on *International Conference on*, pp. 90-94, 2010.

[13] S. Azzounzi., M., Benattou, and M.E.H Charaf," A temporal agent based approach for testing open distributed systems." on *Computer Standards & Interfaces, vol 40, pp.* 23-33.2015.

[14] R. Loffler, B. Guldali, & S. Geisen, (2010). Towards Model-based Acceptance Testing for Scrum. *Software technik-Trends, GI.*

[15] S. Iqbal, M. Khalid, M. N. A. Khan, "A Distinctive Suite of Performance Metrics for Software Design", International Journal of Software Engineering & Its Applications, vol. 7, no. 5, (2013).

[16] S. Iqbal and M. N. A. Khan, "Yet another Set of Requirement Metrics for Software Projects", International Journal of Software Engineering & Its Applications, vol. 6, no. 1, (2012).

[17] M. Faizan, S. Ulhaq, M. N. A. Khan, "Defect Prevention and Process Improvement Methodology for Outsourced Software Projects", Middle-East Journal of Scientific Research, vol. 19, no. 5, (2014), pp. 674-682.

[18] M. Faizan, M. N.A. Khan, S. Ulhaq, "Contemporary Trends in Defect Prevention", A Survey Report. International Journal of Modern Education & Computer Science, vol. 4, no. 3, (2012).

[19] K. Khan, A. Khan, M. Aamir and M. N. A. Khan, "Quality Assurance Assessment in Global Software Development" World Applied Sciences Journal, vol. 24, no. 11, (2013).

[20] M. Amir, K. Khan, A. Khan, M. N. A. Khan, "An Appraisal of Agile Software Development Process", International Journal of Advanced Science & Technology, vol. 58, (2013).

[21] M. Khan and M. N. A. Khan, "Exploring Query Optimization Techniques in Relational Databases", International Journal of Database Theory & Application, vol. 6, no. 3, (2013).

[22] M. N. A. Khan, M. Khalid and S. UlHaq, "Review of Requirements Management Issues in Software Development", International Journal of Modern Education & Computer Science, vol. 5, no. (1), (2013).

[23] M. Umar and M. N. A. Khan, "A Framework to Separate NonFunctional Requirements for System Maintainability", Kuwait Journal of Science & Engineering, vol. 39, no. 1 B, (2012), pp. 211- 231.

[24] M. Umar and M. N. A. Khan, Analyzing Non-Functional Requirements (NFRs) for software development. In IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS), (2011), pp. 675-678).

[25] M. N. A. Khan, C. R. Chatwin and R. C. Young, "A framework for post-event timeline reconstruction using neural networks", digital investigation, vol. 4, no. 3, (2007), pp. 146-157.

[26] M. N. A. Khan, C. R. Chatwin and R. C. Young, "Extracting Evidence from File system Activity using Bayesian Networks", International journal of Forensic computer science, vol. 1, (2007), pp. 50-63.

[27] M. N. A. Khan, "Performance analysis of Bayesian networks and neural networks in classification of file system activities", Computers & Security, vol. 31, no. 4, (2012), pp. 391-401.

[28] M. Rafique and M. N. A. Khan, "Exploring Static and Live Digital Forensics: Methods, Practices and Tools", International Journal of Scientific & Engineering Research, vol. 4, no. 10, (2013), pp. 1048-1056.

[29] M. S. Bashir and M. N. A. Khan, "Triage in Live Digital Forensic Analysis", International journal of Forensic Computer Science, vol. 1, (2013), pp. 35-44.

[30] A. Sarwar and M. N. A. Khan, "A Review of Trust Aspects in Cloud Computing Security", International Journal of Cloud Computing and Services Science (IJCLOSER), vol. 2, no. 2, (2013), pp. 116-122.

[31] A. H. Gondal and M. N. A. Khan, "A review of fully automated techniques for brain tumor detection from MR images", International Journal of Modern Education and Computer Science (IJMECS), vol. 5, no. 2, (2013), pp. 55.

[32] A. Zia and M. N. A. Khan, "Identifying key challenges in performance issues in cloud computing", International Journal of Modern Education and Computer Science (IJMECS), vol. 4, no. 10, (2012), pp. 59.

[33] K. U. Rehman and M. N. A. Khan, "The Foremost Guidelines for Achieving Higher Ranking in Search Results through Search Engine Optimization", International Journal of Advanced Science and Technology, vol. 52, (2013), pp. 101-110.

[34] M. Khan and M. N. A. Khan, "Exploring query optimization techniques in relational databases", International Journal of Database Theory & Application, vol. 6, no. 3, (2013).

[35] R. Shehzad, M. N. KHAN and M. Naeem, "Integrating knowledge management with business intelligence processes for enhanced organizational learning", International Journal of Software Engineering and Its Applications, vol. 7, no. 2, (2013), pp. 83-91.

[36] S. U. Haq, M. Raza, A. Zia and M. N. A. Khan, "Issues in global software development: A critical review", Journal of Software Engineering and Applications, 4(10), 590, 2015.

[37] A. S. Shah, M. N. A. Khan and A. Shah. An appraisal of off-line signature verification techniques. *International Journal of Modern Education and Computer Science, 7*(4), 67-75, 2015.

[38] A. Zia and M. N. A. Khan, "A Scheme to Reduce Response Time in Cloud Computing Environment", International Journal of Modern Education and Computer Science (IJMECS), vol. 5, no. 6, (2013), pp. 56.

[39] M. Tariq and M. N. A. Khan, "The Context of Global Software Development: Challenges, Best Practices and Benefits", Information Management & Business Review, vol. 3, no. 4, (2011).

[40] A. Shahzad, M. Hussain and M. N. A. Khan, "Protecting from Zero-Day Malware Attacks", Middle-East Journal of Scientific Research, vol. 17, no. 4, (2013), pp. 455-464.
[41] A. A. Khan and M. Khan, "Internet content regulation framework", International Journal of U-& EService, Science & Technology, vol. 4, no. 3, (2011).
[42] K. Ullah and M. N. A. Khan, "Security and Privacy Issues in Cloud Computing Environment: A Survey Paper", International Journal of Grid and Distributed Computing, vol. 7, no. 2, (2014), pp. 89-98.
[43] A. A. Abbasi, M. N. A. Khan and S. A. Khan, "A Critical Survey of Iris Based Recognition Systems", Middle-East Journal of Scientific Research, vol. 15, no. 5, (2013), pp. 663- 668.
[44] M. N. A. Khan, S. A. Qureshi and N. Riaz, "Gender classification with decision trees", Int. J. Signal Process. Image Process. Patt. Recog, vol. 6, (2013), pp. 165-176.
[45] S. S. Ali and M. N. A. Khan, "ICT Infrastructure Framework for Microfinance Institutions and Banks in Pakistan: An Optimized Approach", International Journal of Online Marketing (IJOM), vol. 3, no. 2, (2013), pp. 75-86.
[46] A. Mahmood, M. Ibrahim and M. N. A. Khan, "Service Composition in the Context of Service Oriented Architecture", Middle East Journal of Scientific Research, vol. 15, no. 11, (2013).
[47] M. A. Masood and M. N. A. Khan, "Clustering Techniques in Bioinformatics", I. J. Modern Education and Computer Science, vol. 1, (2015), pp. 38-46.
[48] Ur Rehman, T., Khan, M. N. A., & Riaz, N. (2013). Analysis of Requirement Engineering Processes, Tools/Techniques and Methodologies. International Journal of Information Technology and Computer Science (IJITCS), 5(3), 40.
[49] Ahmed, R., & Khan, M. N. A. (2013). An Analytical Review of Stereovision Techniques to Reconstruct 3D Coordinates. International Journal of Information Technology and Computer Science (IJITCS), 5(7), 80.

Authors' Profiles

Muhammad Fraz Malik is pursuing for MS degree in Software Engineering at Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad. His research areas include software engineering and data mining techniques.

M.N.A. Khan obtained D.Phil. degree from the University of Sussex, UK. His research interests are in the fields of software engineering, cyber administration, digital forensic analysis and machine learning techniques.