SYNCHROINFO JOURNAL

www.media-publisher.eu/synchroinfo-journal

SOFTWARE TESTING AND ITS ASPECTS

Conrad Onesime Oboulhas Tsahat [1], **Ngoulou A. Ndzeli** [2]

[1] Ecole Nationale Supérieure Polytechnique, Université Marien NGOUABI, Republic of Congo,
Teacher-researcher, Assistant professor;
oboulhas@yahoo.fr

[2] Ecole Nationale Supérieure Polytechnique, Université Marien NGOUABI, Republic of Congo
Teacher-researcher, Assistant;
becker20000@yahoo.fr

ABSTRACT

The software testing topic is becoming more and more popular. This article discusses the software testing concept, software testing types, how it works and where it is applied. Software testing is the most important phase of the software development life cycle, so this article is about ensuring the quality of all software applications types by executing certain types of testing methods and streamlined software testing processes. The object of the work is the software testing process. The subject of the research is software testing and its life cycle. The purpose of the work is to review the software testing concept, testing types, software testing capabilities, software testing stages, as well as current state presentation of issue. The theoretical method was chosen as the research method. In this paper, an optimized testing process is considered, which considers all testing life cycle stages and also considers the testing types. The article describes the current state of issue. The work objectives have been achieved. The assigned tasks have been completed.

DOI: 10.36724/2664-066X-2024-10-1-2-7

Received: 28.11.2023
Accepted: 12.01.2024

Citation: Conrad Onesime Oboulhas Tsahat, Ngoulou A. Ndzeli, "Software testing and its aspects," *Synchroinfo Journal* **2024**, vol. 10, no. 1, pp. 2-7.

Licensee IRIS, Vienna, Austria.

KEYWORDS: *software testing, software, functional testing, performance and security testing, software testing methods, software testing life cycle, software development life cycle.*

1 Introduction

From the early 1950s, when the first programs were written to the present day, the cost of software products has skyrocketed. And if until the beginning of the 70s there was obvious optimism in the programmers work assessment, then from the beginning of the 70s the crisis began to manifest itself more clearly affecting both the scientific research spheres and production. This crisis was caused both by the programs complexity that solve real production problems and the decrease in these programs reliability. No longer rumors but articles and official reports about the "failures" of projects due to programmer errors and not equipment failures. Thus, one error in the Fortran program caused the failure of the attempt to launch the first American research vessel to Venus which led to several people death in medical institutions and to aircraft crash. As a result, the question arose that something needs to be changed in the programming itself. This is how software testing was born [1-4].

Various testing types are used including black box testing, white box testing, state-based testing, security testing, acceptance testing, acceptance testing, system testing, alpha and beta testing, tuning, verification, and validation. Based on the research and studies that have been done, in this article they have all been divided into three types of high-level tests: functionality, performance, and security.

1.1 Setting a common goal

The work object is the software testing process.

The work purpose is to review the software testing concept, testing types, software testing stages, as well as presentation of issue current state.

1.2 Research problem statement

To achieve this goal, the following tasks were put forward:

To study sources given topic on the Internet;

Consider the software testing concept, highlight the software testing types, consider the purpose.

2 Software testing

2.1 The software testing concept

First, we need to understand what testing is and how the term "program testing" differs from the term "program debugging". Obviously, both debugging and testing are activities aimed part at improving the software products reliability. Despite the fact that the judging problem about correctness of the program has existed for a long time and many scientists have dealt with it, until now for any test problem, one can hear the following statements like “Testing is checking process of program correctness and demonstrating that any program does not contain errors ". However, no programmer can prove that a program is completely error-free. I know it's possible. It is the test "definition" describes the opposite of what the term "test" actually means.

Software testing is software product testing exploratory process with a final tests set chosen in a certain way to verify the correspondence between the program actual behavior and its expected behavior (ISO/IEC TR 19759:2005). .

The testing purpose is to verify the software to the requirements, to give confidence in the quality of the software and to look for obvious errors in the software that need to be identified before the program user discovers them.

Software testing is performed for the following purposes:

1. Compliance check.

2. Identify problems at an early development stage and prevent the increase in the product cost.

3. Use cases discovery that were not foreseen during development. And also look at the product from the user's point of view.

4. Increase loyalty to the company and products. The identified shortcomings negatively affect the users trust.

Here are the seven testing principles.

1) Testing shows the defects presence. Testing only reduces the defects likelihood in the software but does not guarantee their absence.

2) Exhaustive testing is not possible. A complete test with all input combinations of data, results, and assumptions is physically impossible (except in trivial cases).

3) Early testing. Testing should start early in the software development life cycle in order to detect defects as early as possible.

4) Defects accumulation. Most of the shortcomings lie in the limited number of modules.

5) Pesticides paradox. If you repeat the same test scenario over and over again at some point this tests series will stop finding new defects.

6) Tests are context dependent. Testing is done differently depending on the context. For example, security-critical software is tested differently than news portals.

7) The delusion that there are no mistakes. The defects absence during testing does not mean that the product is ready for release. The system should be easy to use and meet its expectations and needs.

Test procedure:

1) Product analysis.

2) Work with requirements.

3) Testing strategy development.

4) Test documents creation.

5) Prototype testing.

6) Basic test.

7) Stabilization.

8) Operation.

The actual result deviation from the expected one is called a defect and also an error.

A bug report is a document that reports that there are no defects in a component or system that could cause the component or system to fail to perform its intended function.

The defect life cycle is shown in Figure 1.

New (New) - Testers find errors, localize them and enter them into a special system where error reports are stored. From this moment the bug officially begins to exist. Rejected (Rejected) or Assigned. Rejected (Rejected) - A comment from a programmer or manager about the reason for reject-a (rejected). This may be a low-quality defect description. The defect already exists (duplicate), the defect cannot be reproduced, etc. Some errors may also occur because they are no longer relevant to the client. Then the tester either closes the defect (Closed) or completes the comments on this defect and transfers the defect to the Assigned state.

Assigned - the defect has been checked and opened (i.e. the fix is recognized). Fixed (Fixed) - the defect has been fixed and in this state it requires re-checking by the tester. After the tester checks the bug the defect is moved to the Closed state if the bug is fixed.

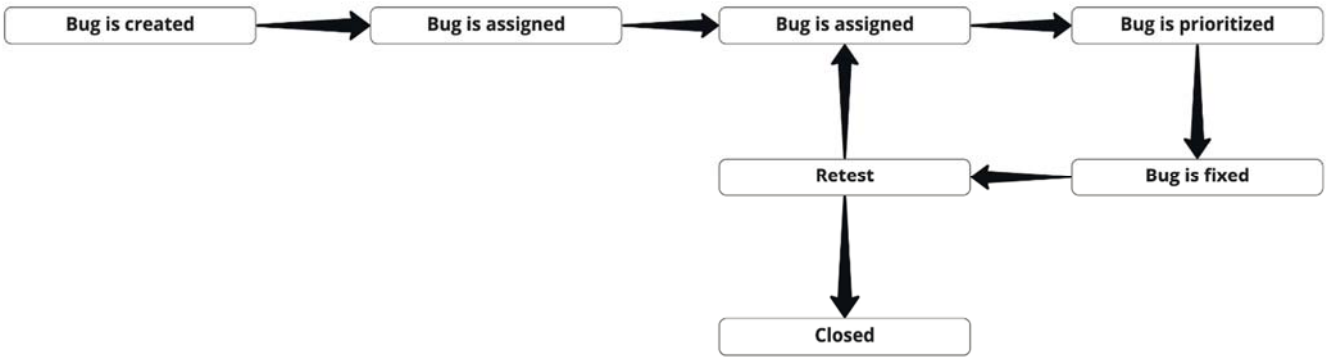

Fig. 1. Defect life cycle

2.2 Software testing types

There are different types of programming according to studies and research, such as black box, white box, gray box, regression, recommendation, usability, performance, modularity, system, integration, security, smoke, health and object-oriented testing, etc. It is not possible to implement all software testing types, so a certain amount of time is always spent on testing. Functional testing is very common and quite a lot of research has been done in the past so it is only in rare cases that a website goes down due to lack of functional testing. This part of the article focuses on the most important testing methods such as functionality "F" [5], performance testing "P" [6-8], and security testing "S" [9-10]. The correct tests combination must be included from all heads F, P and S.

Functionality is primarily an software testing aspect that ensures the software quality. Verification and validation are used for static and dynamic testing, respectively. Static testing checks all kinds checks and passes. Dynamic testing or actual testing includes all functional and non-functional tests.

A) Functional testing.

The main quality factor in software is the conformity of the required function and its behavior. Part of software function is related to the external behavior which basically determines all kinds of user usage. High-level software development is carried out in such way that the customer is satisfied in the early design stages and development. There are different types of functional testing that can be performed at different testing levels such as unit testing, integration testing (top-down testing and bottom-up testing), and system testing. There are many tests at different testing levels such as black box testing, white box testing, gray box testing, regression testing, fog testing, use case testing, certification, smoke testing and feasibility testing. -path, acceptance testing, alpha testing, beta testing, etc. The main functional testing types are briefly described in Table 1.

Table 1

Description of functional testing types

Type of testing	Description
Modular testing	The lowest testing level is mostly done by the developer to test the code unit.
Integrated testing	It is done to check communication between different modules to make sure that data is being passed correctly between different components. It is done on a top-down or bottom-up basis.
System testing	The system is tested as a whole to ensure that it behaves or functions as intended and as specified in the requirements document. Regression testing is done to make sure that nothing is broken in the system after bug fixes and bug testing. General testing by Smoke and Sanity is done to make sure all links and features work and the environment is stable.
White and black box testing	Black box testing is done to ensure that the application output is correct for all the different types of positive and negative inputs. White box deals with internal code processing to ensure that no redundant code is written in the software.

B) Performance testing.

This is a type of non-functional testing that tests the software performance under all fair and unfair conditions.

It includes all time-related parameters such as load time, access time, execution time, execution time, etc. It also includes results, failure rate, mean time between failures and overall program reliability. The most popular testing types in performance testing are stress testing and load testing.

Stress testing is done to determine upper limits and understand system throughput. The final boot is given to the application to confirm the system reliability.

Load testing is called endurance testing. This test is carried out to determine if the system can withstand the expected continuous load.

C) Security testing.

According to current scenarios, flood and buffer attacks are the most common. In an object-oriented system design considerations include error handling. Some other design problems such as communication and trust issues, unreliable communication channels, unstructured or missing access control mechanisms, listening lack, incorrect and timely logging and organizational errors also lead to security risks. The program is required to learn security features such as strong authentication, cryptography and access control as well as some other security mechanisms.

Gary McGraw [11] suggests: “Pleasibility is a bug that an attacker can exploit. Software security testing is critical to protecting the information, services, skills and resources of attackers as well as reducing the potential protection cost.”

a) security testing generally follows two approaches types;

b) software testing in relation to the software functionality;

A risk-based approach requires caution from attackers.

A security penetration test is a test in which security evaluators attempt to bypass the system security features based on their structure understanding and system implementation.

Another type called Fuzz Testing was introduced by Barton Miller [12] from Wisconsin University in 1988. This is a web testing technique that automatically generates false, random and unexpected data to determine how a program will respond. It is good for software testing where input has no control over predefined data. This testing method is only used to find simple programming features but not complex code.

Now, to summarize the above testing sections Table 2 lists all testing types.

Table 2

All testing types description

Testing type	Method
Functional Testing	Black Box, White Box, All Pair Testing, State Transition Tables, Decision Tables, Model Based Testing, Usage Based Testing, Pretesting, Specification Based Testing, Regression Testing, Smoke Testing, Health Testing.
Performance Testing	Load, stress, endurance and configuration testing.
Security Testing	Static Analyzer, Brute Force Attack, SQL Injection and Cross Site Scripting (XSS), Penetration Testing and Fuzz Testing.

5 Conclusion

The software testing process is considered, the software testing theoretical concept is considered, the testing types are considered and the issue current state is outlined in this paper. The work goals and objectives have been achieved.

Since it is always a goal, software testing research can be done in addition to this article to suggest a common test environment and methods to support functional testing, performance testing and security testing for a development environment and other platforms that use some algorithms without the tools use in the shortest possible time.

REFERENCES

[1] R. A. Khan, S. U. Khan, H. U. Khan and M. Ilyas, "Systematic Literature Review on Security Risks and its Practices in Secure Software Development," *IEEE Access*, vol. 10, pp. 5456-5481, 2022, doi: 10.1109/ACCESS.2022.3140181.

[2] E. Enoiu, G. Tukseferi and R. Feldt, "Towards a Model of Testers' Cognitive Processes: Software Testing as a Problem Solving Approach," *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, Macau, China, 2020, pp. 272-279, doi: 10.1109/QRS-C51114.2020.00053.

[3] F. Cacciotto, T. Fulcini, R. Coppola and L. Ardito, "A Metric Framework for the Gamification of Web and Mobile GUI Testing," *2021 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, Porto de Galinhas, Brazil, 2021, pp. 126-129, doi: 10.1109/ICSTW52544.2021.00032.

[4] V. Tomar, M. Bansal and P. Singh, "Regression Testing Approaches, Tools, and Applications in Various Environments," *2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST)*, Delhi, India, 2022, pp. 1-6, doi: 10.1109/AIST55798.2022.10064753.

[5] J. Brown and J. Barkley, "Built-In Test Selection Methodology for Optimal Reliability Fault Coverage," *2021 Annual Reliability and Maintainability Symposium (RAMS)*, Orlando, FL, USA, 2021, pp. 1-5, doi: 10.1109/RAMS48097.2021.9605731.

[6] S. Hooda, V. Lamba, S. Kaur, V. K. Sharma, R. Kumar and V. Sood, "Impact of Software Quality on "GA-FC" Software Testing Technique," *2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES)*, Greater Noida, India, 2022, pp. 142-147, doi: 10.1109/CISES54857.2022.9844275.

[7] D. Ravalika, R. Pitchai and C. M. Babu, "Improving the Quality of Software Solutions using Genetic Algorithm," *2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS)*, Trichy, India, 2023, pp. 998-1002, doi: 10.1109/ICAISS58487.2023.10250475.

[8] C. Pan, J. You and Y. Gao, "AI Software Reliability: Concepts and Related Domains," *2023 2nd International Conference on Artificial Intelligence and Intelligent Information Processing (AIIIP)*, Hangzhou, China, 2023, pp. 287-292, doi: 10.1109/AIIIP61647.2023.00061.

[9] E. R. Heymann and B. P. Miller, "Software Security for the People: Free and Open Resources for Software Security Training," *IEEE Security & Privacy*, vol. 20, no. 2, pp. 88-95, March-April 2022, doi: 10.1109/MSEC.2022.3142336.

[10] D. D. Yao et al., "Being the Developers' Friend: Our Experience Developing a High-Precision Tool for Secure Coding," *IEEE Security & Privacy*, vol. 20, no. 6, pp. 43-52, Nov.-Dec. 2022, doi: 10.1109/MSEC.2022.3159481.

[11] B. Arkin, S. Stender and G. McGraw, "Software penetration testing," *IEEE Security & Privacy*, vol. 3, no. 1, pp. 84-87, Jan.-Feb. 2005, doi: 10.1109/MSP.2005.23.

[12] B. P. Miller, M. Zhang and E. R. Heymann, "The Relevance of Classic Fuzz Testing: Have We Solved This One?," *IEEE Transactions on Software Engineering*, vol. 48, no. 6, pp. 2028-2039, 1 June 2022, doi: 10.1109/TSE.2020.3047766.