Development and Classification of Computer Software Testing Technology

Chenchen Pi
School of Computer Information Engineering, Jiangxi Normal University, Nanchang, Jiangxi, 330022

Abstract. With the rapid development of computer technology, software is more and more used by people, but serious economic losses caused by software errors also occur from time to time, so the quality and reliability of software have become issues that must be paid attention to at present. Software testing, as the only method to verify whether the software can complete its function, has attracted people's attention. Based on this, this article has studied the development and classification of computer software testing technology. First, it gives an overview of software testing, introduces the definition, purpose, principles, and processes. Then analyzes the development of software testing technology, including its development process, current status, and future development trends, and finally classifies software testing from three aspects. This article has a certain significance for comprehensive understanding of software testing.

1. Introduction
With the development of China's IT industry and the maturity of the software market, people's expectations for software functions are getting higher and higher, and software quality, performance, and reliability are gradually becoming the focus of people's attention [1]. As an important part of software survival cycle, software testing is getting more and more attention. Software testing is the only effective way to verify whether the software can perform the desired function. Testing is not just limited to one stage in software development, it has begun throughout the entire software development process, the earlier the test is performed, the more the overall software development cost will decrease [2].

2. Software testing overview
2.1. Definition of Software Testing
After many years of software development practice, the significance of software testing has gradually been widely recognized. However, there are different views on the basic concept of what is software testing. W. Hetzel pointed out in 1973 that testing is the process by which a program or system can complete a specific task to build confidence. The software testing standard definitions proposed by the IEEE in 1983 for software testing are: "The process of running or measuring a system using manual or automatic means, the purpose of which is to check whether it meets the specified requirements or understand the difference between the expected result and the actual result. "GJMyers takes a different view. He believes that" program testing is the process of executing a program to find errors. "So far, for software testing what is more complete in all definitions is that software testing is to analyze a software
item to find the difference between video memory and required conditions and evaluate the characteristics of the software [3].

2.2. Description of Software Test
Testing is an important part of the software development process, and it is used to confirm whether the quality or performance of a program meets some requirements put forward before development. The purpose of software testing is to confirm the quality of the software. The second is to provide information, such as feedback to developers or program managers, and information for risk assessment; The three software tests are not only testing the software product itself, but also the process of software development [4]. If a lot of problems are found after the development of a software product, this indicates that the software development process is likely to be flawed.

2.3. Purpose of Software Testing
Software testing is a program execution process, the purpose is to find and correct errors in the tested software as much as possible, to improve the reliability of the software. It is a very important and very complicated work in the software life cycle, and it is a software reliability assurance which has extremely important significance. In the case that the current formal methods and program correctness verification technologies are not yet expected to become practical methods, software testing will still be an effective method for software reliability assurance for a considerable period of time in the future [5]. The overall goal of software project is to make full use of limited human and material resources to complete software development projects efficiently and with high quality. Insufficient testing is bound to bring the software into operation with some undisclosed hidden errors, which will mean greater danger users bear. Excessive testing will waste many valuable resources. Later in the test, even if an error is found, the price is too high. A famous quote from EWDijkstra illustrates this: "Program testing can only indicate the existence of errors and cannot indicate that the error does not exist." It can be seen that the test is to make the defect contained in the software lower than a certain value, so to maximum the rate of output and invested. The overall goal is: to ensure software quality.

2.4. Principles of Software Testing
Bill Hetzel talks about six principles in his book "The Complete Guide to Software Testing". The so-called testing principles are the concrete manifestation of internal laws in the testing process and have been recognized. These principles can help us understand the meaning of testing.

- Principle 1: Exhaustive testing is impossible.
- Principle 2: Testing is creative but difficult.
- Principle 3: Testing is designed to prevent errors.
- Principle 4: Testing is risky.
- Principle 5: Testing needs to be planned.
- Principle 6: Testing needs independence.

2.5. Software Testing Process
Test process means the series of preparation, execution, and analysis processes from software test start to software test end [6]. Generally speaking, software testing begins when the project is established, and it must go through the following main links before and after: requirements analysis → test plan → test design → test environment construction → test execution → test recording → defect management → software evaluation → RTM. Before elaborating related issues, you need to clearly define the division of labor. Generally speaking, requirements analysis, test case writing, test environment construction, and test execution belong to the test developer. While test execution and defect submission belong to the work scope of ordinary testers, and the test leader is responsible for the tracking, implementation, management, etc. of all aspects of the entire test. Software test is more than just a process of the internal work flow of the department, it must also formulate the process of interface work with external departments such as the development department and the demand department. Formulating a reasonable
software testing process is a deep learning. It requires the developer to have a wealth knowledge of software testing theory, software testing execution experience, management experience, and communication skills, and requires many testers to verify and improve after a long period of practice.

3. Development of software testing

3.1. Development Process of Software Testing
Software testing is accompanied by the emergence of software. Early software testing was relatively narrow. And equivalent to "debugging". Developers completed this part of the work themselves. Usually, testing is performed only after the code is formed and the product is basically completed. The investment is very small. Until 1957, software testing became an activity to find software defects. Due to the lack of software engineering concepts, testing is still a matter of development. In 1972, the first formal meeting of software testing was held at the University of North Carolina. In 1975, John Good Enough and Susan Gerhart published an article on "Toward a Theory of Test Data Selection" on IEEE, and software testing was only identified as a research direction. In 1979, "The Art of Software Testing" written by Glen ford Myers's is the first important monograph in the field of testing. In this book, Myers and colleagues define software testing as "a test is a program executed to find errors or systematic process".

By the 1980s, software testing was no longer a process of simply finding errors, but also included the content of software quality evaluation, including IEEE (Institute of Electrical and Electronic Engineers) standards, American ANSI (American National Standard Institute) standards, and ISO (International Standard Organization) were successively formulated. In 1983, Bill Hetzel pointed out in the "Complete Guide of Software Testing" that "testing is to evaluate the properties of a program or system as for any kind of activity, testing is a measure of software quality." In the 1990s, testing tools began to prevail. In 2002, testing was defined as "testing is to measure and improve the quality of the software under test. It is the entire life cycle process of engineering design, implementation and maintenance". In the past 20 years, software testing technology has made great breakthroughs with the rapid development of computer and software technology. Test models including V model and W model have been summarized one after another [7]. At the same time, the emergence of the TMM (Testing Maturity Model) concept marks the improvement of the testing process. A large number of excellent management of software testing tools such as unit testing, automated testing, load testing and stress testing emerged.

3.2. Status of Software Testing at Home and Abroad
(1) Software testing plays an important role in software companies. At Microsoft, there are far more test engineers in a project team than coding engineers, and they also spend more time testing than coding.

(2) Software testing theoretical research is booming. Various annual testing technology conferences are held every year, and a large number of software testing research papers are published, leading the international trend of software testing theoretical research.

(3) The software testing market is booming. There are some professional companies in the United States that develop software testing standards and testing tools. MI Compurware, MaCabe, and Rational are all well-known software testing tool providers. The testing tools produced by them have occupied the international market.

(4) China’s testing technology started in the "Sixth Five-Year Plan" and gradually developed with the study of software engineering. In 1990, the national software evaluation center of China was established, and testing services were gradually launched.

Because of its late start, there is a large gap between developed countries in both software testing theoretical research and testing practice [8]. This is mainly reflected in the lack of employees, testing services and technical research on software productization testing. However, with the vigorous development of China's software industry and the emphasis on software quality, software testing is increasingly valued by people, and software testing is gradually becoming an emerging industry.
3.3. Development Trends of Software Testing
Analyzing the development of testing at home and abroad today, we can see the following trends:

The testing work is moving forward, not only unit testing, integration testing, system testing and acceptance testing. Testing technology for the accuracy and completeness of requirements, and testing technology for system design will become new research hotspots.

Software architects, development engineers, QA personnel, and test engineers will be better integrated. They are partners rather than antagonistic relationships, because their work promotes each other and learns from each other [9]. At the same time, test engineers will also intervene as early as possible for the entire project, corresponding test methods must be developed during the software definition phase.

The testing profession will be fully respected. Development and testers are both contradictory and unified. The previous view that "no ability then to test" has been replaced by the current view that "only high-level developers can do the test work". Setting up an independent software testing department will become the consensus of more and more software companies. The testing department will exist as an important independent department like QA.

Testing outsourcing services will grow rapidly. Like software development outsourcing, software testing outsourcing will become a global trend, and you can use professional testing expert teams and institutions to test your own products and save testing costs.

4. Classification of software testing
From different perspectives, there can be different division methods. The classification of tests is to better clarify the test process, understand what work the test must complete, and try to achieve a comprehensive test.

4.1. Whether to Execute the Program
According to whether the software under test needs to be executed, it can be divided into static testing and dynamic testing. Static testing refers to the process of not actually running the software under test, but only statically checking for possible errors in program code, interface, or documentation. Including code testing, interface testing, and document testing. For code testing, the main test is whether the code meets the corresponding standards and specifications. For interface testing, the main test is whether the actual interface of the software matches the description in the requirements. For document testing, the main test is whether the user manual and the requirements description meet the actual requirements of the user. Dynamic testing refers to the process of actually running the program under test, entering the corresponding test data, and checking whether the actual output results and the expected results are consistent. Therefore, the only criterion we judge a test to be dynamic is to see if the program is running.

4.2. According to The Test Method
4.2.1. White box test. White box test is also called structural test or logic-driven test. It is to know the internal working process of the product. It can be tested to test whether the internal action of the product is performed normally according to the specifications. Whether each path in the program can work correctly according to the predetermined requirements, regardless of its function. The main methods of white box testing are logic driving, basic road testing, etc., which are mainly used for software verification. White box testing is based on the program internal logic design test cases. A commonly used technique is logical coverage, that is, the degree of coverage of program logic when running the tested program with test data. There are six main coverage standards: statement coverage, decision coverage, condition coverage, decision / condition coverage, conditional combination coverage, path coverage [10].

White-box testing is testing through the source code of the program without using the user interface. This type of testing needs to find the shortcomings or errors of the internal code in algorithms, overflows, paths, conditions, etc. from the code syntax, and then correct it. When testing, you can understand the
structure of the object under test, and you can refer to the test work of the content of the code under test. It needs to know the design structure and specific code implementation inside the program, and use this as a basis to design test cases. The advantage is to know where the designed test cases are ignored at the code level. Its advantage is to help software testers increase the coverage of the code, improve the quality of the code, and find hidden problems in the code. At the same time, white box testing also has cons: 1) There will be many different paths for the program to run, it is impossible to test all the running paths; 2) The test is based on the code. It can only test whether the developer did right, but can't know whether the design is correct or not. It may miss some functional requirements; 3) When the system is large, the test overhead will be very large.

4.2.2. **Black box test.** Black box testing, as its name implies, treats the system under test as a black box, taking input from the outside, and then outputting it. The entire test is based on the requirements document to see if it can meet all the requirements in the requirements document. Black box testing requires the tester cannot use knowledge or experience related to the internal structure of the system under test, and it is suitable for testing the function of the system. Black box testing is also called functional testing or data-driven testing. It is a function that should be available in known products. The entire software or a software function is strictly tested to check whether each function can be used normally without checking the source code of the program or clearly understanding the specific source code of the software or a software function. Testers understand how the software works by entering their data and looking at the output results. When testing, consider the program as a black basin that cannot be opened. Without considering the internal structure and internal characteristics of the program at all, the tester performs the test on the program interface. It only checks whether the program function is properly used according to the requirements specification. Whether the program can properly receive input data saws to produce correct output information, and maintain the integrity of external information (such as databases or files). Black box testing methods mainly include equivalence class division, boundary value analysis, cause-effect diagrams, error speculations, etc. which are mainly used for software confirmation testing. The "black box" method focuses on the external structure of the program, ignoring the internal logical structure, and tests the software interface and software functions. The "black box" method is exhaustive input testing. Only by using all possible inputs as test cases can we find all errors in the program in this way. In fact, there are an infinite number of test cases. People not only have to test all legal inputs, but also those that are illegal. Usually testers use not only input data that is sure to produce correct results, but also use challenging input data and input data that may result in errors in order to understand how the software handles various types of data.

The advantages of black box testing are: 1) It is relatively simple and does not need to know the code and implementation inside the program; 2) It has nothing to do with the internal implementation of the software; 3) From the user's perspective, it is easy to know which functions users will use and what problems they will encounter; 4) Based on the software development documentation, you can also know which functions in the documentation the software has implemented; 5) It is more convenient when doing software automated testing. The disadvantages of black box testing are: 1) It is impossible to cover all the code, and the coverage rate is low, which can only reach 30% of the total code amount; 2) Reusability of automated testing is low.

4.2.3. **Gray box test.** Gray-box testing: Gray-box testing passes user interface testing just like black-box testing, but testers already have some understanding of how the software or the source code program of a software function is specifically designed, and even read part of the source code, so testers can purposely perform some certain conditions / function tests. The significance of this is that if you know the internal design of the product and the user interface of the product through the user interface will allow you to test its performance more effectively and deeply from the user interface. The gray box test is between the white box test and the black box test. It can be understood that the gray box test focuses on the correctness of the output for the input and also focuses on internal performance, but this attention is not as detailed and complete as the white box, just to judge the internal running status through some
symbolic phenomena, events, and signs. Sometimes the output is correct, but the internal is actually wrong. This is very common. It will be very inefficient if pass the white box test every time to operate, so it is necessary to adopt such a gray box method. Gray box testing combines the elements of white box test box and black box test, which considers the client, specific system knowledge and operating environment. Testing consists of methods and tools that are drawn from the environment in which the application's internal knowledge box interacts. They can be used for black box testing to enhance the efficiency of testing, error detection and error analysis.

Grey-box testing involves input and output, but design tests with information about code and program operations, etc. that are usually outside the tester's vision.

4.3. According to the test phase

4.3.1. Unit testing. Unit testing is the testing of the basic constituent units of software, such as a module, a process, etc. It is the most basic and one of the most important part of software dynamic testing. Its purpose is to verify the basic constituent units of software because unit tests need to know the details of internal program design and coding, they should generally be completed by programmers rather than testers. They often need to develop test-driven modules and stub modules to assist in completing unit tests. Therefore, the application system is especially important to design a well-designed architecture. Unit testing mainly uses white-box testing methods. Generally, first statically check whether the code conforms to the specification, and then dynamically run the code to check its actual running result. Of course, checking whether the running result is correct is the most basic requirements, we also have to check many items, such as fault-tolerant processing of the program, boundary value processing of the program, etc.

4.3.2. Integration test. Integration test is a test performed during the software system integration process, and its main purpose is to check whether the interface between software units is correct. It integrates modules or other software units into an increasingly larger system according to the integration test plan. Run the system to analyze whether the system is correct and whether the components are in sync. The integration testing strategy mainly has two types: top-down and bottom-up. Unit testing only guarantees the local correctness of the module. The system testing is generally performed after the entire system is completed, and errors are difficult to locate.

4.3.3. Confirmation test. After the integration test is completed, the decentralized development modules are connected to form a complete program. All the problems in the interface between the modules are eliminated. Therefore, the test work enters the final stage-confirmation test. Among the many statements about what is confirmation test, the most rigorous and strictest explanation is to check whether the software developed can be performed according to customer requirements. If this requirement is met, the software issued by the card is deemed to be qualified. Therefore, some software development departments have adopted the confirmation test which is called the qualification test. The confirmation test has the following two steps:
   (1) Confirm test criteria.
   (2) Configuration review.

4.3.4. System test. System testing is actually a comprehensive inspection of each component of the system. Although each inspection has specific goals, all inspections must verify that each part of the system has been properly integrated and can complete the specified functions.

5. Conclusion

With the rapid development of computer application technology, the quality of software directly affects the development process of enterprises, resulting in software development and software testing. Nowadays, because of the continuous expansion of software scale, the complexity of software design
continues to increase, and there are more and more opportunities for software errors or defects in development. At the same time, as people pay more and more attention to software quality, software testing has become more and more important in software development.

References
[1] Zeng Lihong. Analysis software automation test technology and application[J]. Information System Engineering. 2017 (10): 88-92.
[2] Yang Libo. Analysis of the development status of software automation test methods[J]. Wireless Internet Technology. 2017 (04): 56-59.
[3] Ai Di’an. Research and application of software automation test method[J]. China New Telecommunications. 2016 (17): 143-146.
[4] Li Qingnian. Research on computer software testing methods based on multi-platform[J]. Wireless Internet Technology. 2019 (06): 78-82.
[5] Xiang Nan. Software testing technology and test management analysis[J]. Science and Technology Innovation. 2018 (31): 225-228.
[6] Han Tao. Application of software testing strategies and methods[J]. Information recording materials. 2018 (11): 160-163.
[7] Ding Shunying, Li Tianyi, Yu Peng. Research on software testing standardization at home and abroad[J]. Chinese Standardization. 2019 (11): 189-193.
[8] Liu Dongxu, Wang Shanqin. Analysis on the application of software testing process in university software testing studio[J]. Journal of Panzhihua University. 2016 (05): 267-269.
[9] Hu Hai. Discussion on computer software deep development and testing technology[j]. Computer Fans. 2017 (10): 136-140.
[10] Du Juan, Luo Qing, Zhang Quan. Discussion on computer software testing technology and deep development application[J]. Electronic Test. 2017 (19): 98-101.