Analysis of Testing Methods of Large-scale Information Software

Bin Wang, Hang Li

Yunnan Electronic Information Products Inspection Institute, Kunming 650031, Yunnan, China

Abstract: *Software testing is an extremely important work link in software engineering. It runs through the entire life cycle of software engineering to ensure that software development meets requirements and can meet certain software quality requirements. With the continuous expansion of the scale of information systems, simple software testing can no longer meet the requirements of system and software quality assurance. This requires practitioners to gradually begin to think about and improve the testing methods of large-scale information software. For large-scale information software, testing will be embedded in each unit, and the software testing process will be organized in a more macroscopic way of thinking and organization. This article will summarize a series of methods suitable for large-scale information software testing through the research of software testing and the practice of the author. It is hoped that the research in this article can provide some help for the testing of large-scale information software, and also provide help for the precipitation and improvement of the author's own work.*

Keywords: *Information Software, System Testing, Software Engineering*

1. Introduction

With the continuous development of information technology, information systems have gradually been combined with various traditional businesses to form larger-scale and more logically complex information systems and software. The large-scale and complicated information software makes software testing face new challenges. First, the types of software designed by the software system are gradually expanded, from traditional PC or web systems to mobile terminal software, embedded software, and so on. At the same time, the expansion of the software scale has made the scale of demand that the software has to deal with and the source of demand more extensive, which also introduces rapid iteration of user needs. At the same time, the development and organization model of large-scale information software has also changed from traditional team collaboration to cross-domain collaboration and even large-scale parallel development [1]. Under this status quo, traditional manual software testing has encountered a series of bottlenecks, such as insufficient time reserved for testing work and insufficient preparation; the comprehensive quality of testers varies, and the test team is small; The software testing approach is more traditional and cannot cope with high-speed iterations. The emergence of the above-mentioned challenges has forced the testing work to change its mode, and meet the testing needs of large-scale information software through working ideas, organization methods and testing methods that are more suitable for large-scale information software. This article will discuss the necessary testing methods for some large-scale information systems.

2. Pre-planning of intensive testing

Traditional software project managers often underestimate software testing, especially when R&D tasks are tight and R&D schedules are busy, software testing is often ignored. Due to the contempt of software testing by project managers, the share of software testing in the life cycle of software engineering is seriously insufficient. This directly leads to the fact that testers do not have enough time to fully test, nor can they effectively discover the problems of the software system, which in turn affects the quality of the software system. Therefore, in the process of software project management, the pre-planning of testing work should be strengthened, and the testing goals and testing paths should be designed early in the project and included in the project promotion plan [2].

During the test planning process, the following work should be focused.

(1) Clarify the type of project, determine the scale and requirements of the project, and clarify the level of testing to be implemented in this project.

(2) Further clarify the project characteristics, determine whether the test target is system software, application software or embedded software, and set test items and test content in a targeted manner.

(3) Clarify the unit and configuration items, agree on the test environment required by the project, and clarify the tools and resources.

(4) Clarify the test objectives, priorities and test plans of the test work.

(5) Establish the corresponding relationship between the test item and the demand point in the project-related demand document to ensure that testing and R&D can be advanced simultaneously.

Software testing is always a gradual process. Only by establishing a detailed plan linked to requirements as early as possible in advance, can software testing be carried out in an orderly manner during the project advancement process, and ultimately ensure that the quality of large-scale information software meets the design expectations.

3. Synchronous advancement of R&D and testing

In the related theories of software engineering, life cycle theory is often used to explain the software development process. Generally speaking, errors in a software system are not only generated during the coding phase, but may also have occurred during the requirements analysis, outline design, and detailed design phases. Therefore, simply placing the testing work in the development stage or even the final stage of development will inevitably increase the risk of software project advancement. It is also for this reason that testing work should run through the entire software life cycle, from requirements analysis to software design to software implementation, testing work should be added to all links in order to find defects as early as possible and avoid larger problems [3].

3.1. Testing at the software requirement analysis stage

In the requirements analysis stage, engineers mainly capture requirements and form requirements specification documents based on the requirements obtained. At this stage, the test engineer should strictly review the requirements specification documents and test the requirements captured and organized to ensure that the requirements are fully covered, and the requirements are clearly and accurately expressed, and there is no ambiguity. At the same time, the test project needs to be clear at this stage, whether the output requirements specification document can support the test work to be effectively promoted in the future.

3.2. Testing at the software design stage

The testing in the software design phase is essentially oriented to the design results, cooperating with other roles to jointly evaluate whether the design meets the requirements, and whether the design clarifies the relationship between the modules and whether it has good exception handling logic.

3.3. Testing at the software coding stage

The testing at the software coding stage is the testing for the results of software coding in traditional cognition. Under normal circumstances, developers are the actual execution of static tests and unit tests, while testers usually use automated testing tools or test cases to guide developers or perform functional tests on their own. After the coding is completed, a series of automated defect detection frameworks can usually be used for analysis to ensure that the system can automatically trigger testing and optimization during the continuous integration process to ensure reliable results in the coding phase [4].

3.4. Testing of software integration

The integration phase is usually to test the configuration items of the software. In the requirements analysis stage, a series of contents such as software functions and interfaces have been determined, which need to be processed in the software integration stage to ensure that the system has strong consistency and robustness.

3.5. Testing of software joint debugging

Software joint debugging means that after the entire set of software and the system are fully integrated, performance and business tests are performed based on the business logic obtained from the demand analysis, using the real work flow and processing flow. This stage not only requires in-depth testing of the business itself, but also needs to test the overall system capacity, reliability, data processing capabilities, security, etc., to comprehensively evaluate the software quality.

4. Improve the ability and level of automated testing

Automated testing is a testing method that can effectively replace manual completion of many testing tasks that are difficult to manually advance. Reasonable use of automated testing tools to improve automated testing capabilities and levels can effectively improve testing efficiency, shorten development cycles, and save development costs.

4.1. Automatic code defect detection

Although the theory and practice of software testing have been developed for many years, the application of automated testing in most development scenarios is still limited, mainly due to the large scale of the software and frequent changes in requirements. The lack of automated testing tools has threatened development effectiveness and quality. This requires the application of automated testing of software code defects is extremely necessary. By introducing an appropriate code analysis framework, code defect detection rules and mechanisms can be defined in the early stage of development, and automatic defect monitoring mechanisms can be embedded in each link of code writing, warehousing, integration and testing to ensure that the code in each link is safe and of high-quality [5]. This method can effectively reduce the cost of defect repair and improve the efficiency of software development.

4.2. Automatic generation of test documents

In fact, the process of software testing is basically consistent with the principles of all aspects of software engineering, that is, the specification documents of the previous stage are required to be used as the basis and source of work in the latter stage. The same is true for testing. Test requirements analysis, test planning, test design, test execution, test evaluation and other links should output corresponding documents. However, manually sorting and compiling the results of the above-mentioned links consumes the experience and time of testers. Therefore, software technology can be actively used to automatically form test reports at the above-mentioned different test work stages, and output appropriate circulation documents according to the documents [6]. The automatic generation of test documents can effectively improve the efficiency of the test work, and in the case of limited testers, improve the work effectiveness and quality of the test work.

5. Establish a networked test environment

In order to further ensure that the developed system can be well compatible with different types of hardware and software environments, and can withstand different levels and requirements of software evaluation, and to ensure the reliability of the developed system, we should start with the establishment of a networked software testing support environment. And in addition, we should also use the corresponding evaluation environment integrating simulation, testing and evaluation to support the testing work.

First of all, it should be based on the relevant theories and methods of systems engineering, and on the basis of the application of professional test tools and test theories, to conduct rigorous demand mining and analysis for the test work to clarify the core purpose and requirements of the test work. Afterwards, it integrates the design, execution, analysis and evaluation of the test scheme, and completes all the above-mentioned evaluation tasks under a set of platforms to improve efficiency.

Second, it is possible to build a standardized testing support environment that meets the testing objectives as the goal, establish a basic platform for software evaluation, and gradually improve it, and finally form a networked testing environment that meets the overall testing objectives.

Third, we should always keep abreast of the development of related theories and methods of testing, and continuously introduce advanced testing and evaluation technologies in the networked testing support environment to ensure that the testing environment and testing methods are always advanced, reliable and effective. The networked test environment created in this situation can effectively respond to software evaluation requirements of different levels and different purposes, and support system evaluation work of various scales and types [7].

6. Establish a test work evaluation system

For large-scale information systems with rich internal interactions, testing is always a key factor directly related to the success or failure of the entire system. To ensure that the entire set of software and systems can have good software quality, it is necessary to increase investment in testing work. On the one hand, the establishment of testing work evaluation system should be strengthened, and on the other hand, the quality and effectiveness of testing work should be monitored at all times. The method constrains the testing process to ensure that the test is measurable, monitorable, and predictable.

The process of software testing can also be quantified in essence, so a certain quantitative evaluation plan should be established on the basis of the finalized staged work description and evaluation. The difficulty of this work lies in how to quantitatively evaluate the test work. While ensuring the objectivity and authenticity of the quantification, it further ensures that the lighting work is complete [8] so you can try to test the effectiveness. A dynamic measurement standard and method is established between the three points of adequacy and efficiency, and a set of fuzzy decision-making models are constructed through fuzzy mathematics to realize the quantification and evaluation of the software testing process.

7. Conclusion

Although software testing has existed as early as the emergence of software development technology, but limited to the completeness of software engineering and the lag in practice progress, software testing is still a relatively marginal and downstream work in many teams. In fact, to ensure the quality of large-scale information software, software testing should be based on the status of the enterprise team and project characteristics, and targeted improvements in ideas and methods to ensure that software testing can truly penetrate the software under the complex system design requirements. Life cycle, to ensure that software project risks can be controlled within an acceptable range. This article once again puts forward some test methods that can effectively guarantee the quality of large-scale information software, and hope to provide some help to related fields.

References

[1] Han Tao. Application of software testing strategies and testing methods [J]. Information Recording Materials, 2018, 19(11): 97-98.
[2] Wu Zhaoyu, Zhang Yueqin, Yan Hua. Research and application of software testing methods [J]. Journal of Taiyuan University of Technology, 2016, 47(03): 379-383.
[3] Qinghua Ha, Dayou Liu, Xiangheng Shen, Luo Liu. A method for generating test cases for aerospace software based on a demand model [J]. Optics and Precision Engineering, 2016, 24(05): 1185-1196.
[4] Yang Peipei, Zhao Haisheng, Li Zhenxing. Research on practical software testing methods [J]. Computer Applications, 2015, 35(S1): 166-167+173.
[5] Sun Hui. Design and implementation of system testing for a certain software [D]. Beijing University of Posts and Telecommunications, 2015.
[6] Chen Wenbing. Research and application of quality measurement based on software testing [D]. University of Chinese Academy of Sciences (School of Engineering Management and Information Technology), 2015.
[7] Wang Zhenzhen. The preliminary framework of software testing theory [J]. Computer Science, 2014, 41(03): 12-16+35.
[8] Xu Siyan. Design and implementation of software system testing for large-scale command and control information systems [D]. South China University of Technology, 2013.