MAIN ASPECTS OF THE TESTING SOFTWARE SYSTEMS AND COMPLEXES

Postova Svitlana Anatatoliivna,
Candidate of Pedagogical Sciences, Associate Professor, Associate Professor of the Department of Computer Sciences and Information Technologies
Zhytomyr Ivan Franko State University

Melnyk Anna Vitalivna,
Candidate of Pedagogical Sciences, Senior Lecture of the Department of Computer Sciences and Information Technologies
Zhytomyr Ivan Franko State University

Fedorchuk Anna Leonidivna,
Candidate of Pedagogical Sciences, Associate Professor, Associate Professor of the Department of Computer Sciences and Information Technologies
Zhytomyr Ivan Franko State University

Last years the technology of creating software (software) has become the basis of various sections of computer science as a means of overcoming the complexity inherent in modern software systems. Software products are increasingly embedded in various complex real-time systems. Working on these projects requires from software engineers and testers a broad view and mastering of general problems of designing and using systems of a certain purpose and field of use.

A software engineer must participate in the development of requirements for the entire system, as well as learn the application scope of the created set of programs before starting testing and thinking about the functions of the components, their characteristics and tests, the requirements of which the software product will comply with the requirements.

Software testing is the process of detecting errors in the software complex (searching for non-compliance of the system with the requirements), which determines the correctness, completeness and quality of the developed software product. Testing is carried out by an independent group of testers after the product is completed by the developer and before it is handed over to the customer for trial operation [2; 3].

In the broadest sense of the term, testing is the performance of a set of tasks to check the correct functionality of the program. The testing can detect the presence of an error, and debugging can detect the cause of the error, so these two stages of program development "overlap".

Therefore, the main purpose of testing program complexes and their functional components is to identify, register and eliminate defects and errors introduced during the consistent development and implementation of requirements for the functions and characteristics of the program complex.

The tasks of testing and the corresponding phases of testing are presented in Figure 1.

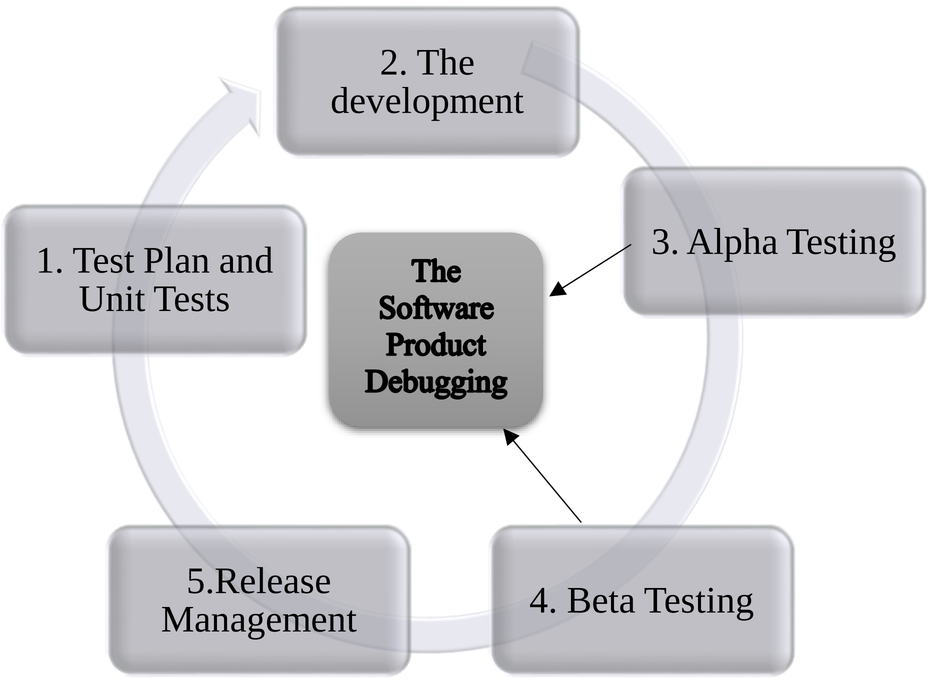

Figure 1. The tasks of testing and the corresponding phases of testing

Thus, the full cycle of quality control of the developed software product includes the following stages:

1. drawing up a test plan (before the system development stage), for existing systems, the test plan is drawn up immediately before the system is handed over for testing;
2. development of unit tests for automated control of compliance of the developed system with the test plan (the stage can be skipped - it is displayed in the test plan);
3. software product development (programming and debugging);
4. preliminary α-testing of the system by developers;
5. β-testing by testers: detection of errors and comments, formulation of proposals (own testing);
6. transfer of tickets to the developer for determining and approving the release / set of changes, setting the release date / making changes [1; 2].

The most important principle in program testing is that this stage should be thought about during the entire period of program development. When creating another fragment of the program, you must have in mind a test that could check the correctness of its operation. If there is no answer to the question of how to test this fragment, it may be necessary to break it into subroutines that are easier to test, or simply rewrite it.

The quality of testing is largely determined not by the number of test runs. The main thing is that each subsequent test run would control something that was not checked in previous runs. The task of testing is to create the most intense mode of operation for the program.

When conducting all tests, it is necessary to have a clear idea about the correct result. The first test can be quite simple. Its main purpose is to check whether the program will work at all. That is why it is also called the "smoke" test. Further complexity of the tests should occur gradually, adding one to the tested elements of the program for each test. If with the help of one test you try to check several subroutines or nodes of the program at once, then when an error occurs, it will be difficult to localize it.

The basic principles of program testing are:

1. Using the principle of protective programming.
2. Testing of boundary conditions (so-called control tests):

   - in the conditional expression, it is necessary to make sure that the branching is performed correctly;
   - it is necessary to provide for checking whether the body of the cycle will be executed the required number of times etc.

The main idea is that when an error occurs, it can be said with a fairly high probability that it is connected precisely with going beyond the limit values. Conversely, if the program works correctly at all extreme values of the test data, it will most likely behave correctly under normal conditions..

3. Analysis of test results. This can be done in several ways: for comparison, calculate the result in another way (for example, on a calculator), use tabular data, etc.
4. Testing individual blocks independently of each other [1].

Thus, the procedure for testing software complexes includes:

• detection of remarks - checking the correctness of system functioning, verification of documentation (analysis of compliance of the system with the technical task, technical project, interface layouts);

• simulation of situations – use of various variants of the sequence of actions performed by the user to check a specific function;

• localization of the detected remark - clarification of the cause of the error; confirmation of the algorithm of actions, which localizes violations in the operation of the system;

• description of the remark - a detailed, formalized textual description of the detected error (may be accompanied by an explanatory illustration);

• formulation of proposals - introduction of proposals to optimize users' work with the system based on the assessment of the usability of the interface and the system as a whole.

All discovered comments and suggestions are recorded by the tester in the ticket register according to standardized parameters: 1) ticket group (digital designation that allows you to assess the level of further changes); 2) the block of the software complex; 3) the section/function; 4) content of the ticket; 5) explanation/example; 6) ticket type; 7) release; 8) date of entry; 9) the employee who registered the ticket; 10) the user who discovered the error; 11) scheduled term for error correction; 12) check mark; 13) responsible for checking; 14) conclusion/note; 15) urgency; 16) importance; 17) reserved fields; etc [1].

After ongoing testing, the tickets are combined into releases of the information system or into separate sets of changes individually or to several sections of the software complex.

During the entire testing period, at the request of the tester, the developer is obliged to provide consulting assistance on the issues of the functioning of the software product. This approach makes it possible to optimize the process of developing a software product and reduce time spent until the release of its fully functional version

References:

1. Avramenko A.S., Avramenko V.S., Koseniuk H.V. Software testing. Educational manual. [Ukrainian: Testuvannia prohramnoho zabezpechennia. Navchalnyi posibnyk]. Cherkasy: ChNU imeni Bohdana Khmelnytskoho, 2017, 284 p.

2. Karpenko M.Iu. Technologies for creating software products and information systems: teaching. Educational manual. [Ukrainian: Tekhnolohii stvorennia prohramnykh produktiv ta informatsiinykh system : navch. posibnyk]. Harkiv : KhNUMH im. O. M. Beketova, 2017, 93 p.

3. Morze N.V. Information systems. Education manual [Ukrainian: Informatsiini systemy. Navch. posibn.] /za sciese ed. N. V. Morze; Morze N.V., Pikh O.Z. Ivano-Frankivsk, «LileiaNV», 2015, 384 p.