Test case

In software engineering, a **test case** is a specification of the inputs, execution conditions, testing procedure, and expected results that define a single test to be executed to achieve a particular software testing objective, such as to exercise a particular program path or to verify compliance with a specific requirement.[1] Test cases underlie testing that is methodical rather than haphazard. A battery of test cases can be built to produce the desired coverage of the software being tested. Formally defined test cases allow the same tests to be run repeatedly against successive versions of the software, allowing for effective and consistent regression testing.[2]

Contents

Formal test cases

In order to fully test that all the requirements of an application are met, there must be at least two test cases for each requirement: one positive test and one negative test.[3] If a requirement has sub-requirements, each sub-requirement must have at least two test cases. Keeping track of the link between the requirement and the test is frequently done using a traceability matrix. Written test cases should include a description of the functionality to be tested, and the preparation required to ensure that the test can be conducted.

A formal written test case is characterized by a known input and by an expected output, which is worked out before the test is executed.[4] The known input should test a precondition and the expected output should test a postcondition.

Informal test cases

For applications or systems without formal requirements, test cases can be written based on the accepted normal operation of programs of a similar class. In some schools of testing, test cases are not written at all but the activities and results are reported after the tests have been run.

In scenario testing, hypothetical stories are used to help the tester think through a complex problem or system. These scenarios are usually not written down in any detail. They can be as simple as a diagram for a testing environment or they could be a description written in prose. The ideal scenario test is a story that is motivating, credible, complex, and easy to evaluate. They are usually different from test cases in that test cases are single steps while scenarios cover a number of steps of the key.[5][6]

Typical written test case format

A test case is usually a single step, or occasionally a sequence of steps, to test the correct behaviour/functionality, features of an application. An expected result or expected outcome is usually given.[7]

Additional information that may be included:[8]

- **Test Case ID** - This field uniquely identifies a test case.
- **Test case Description/Summary** - This field describes the test case objective.
- **Test steps** - In this field, the exact steps are mentioned for performing the test case.
- **Pre-requisites** - This field specifies the conditions or steps that must be followed before the test steps executions.
- **Test category**
- **Author-** Name of the Tester.
- **Automation -** Whether this test case is automated or not.
- **pass/fail**
- **Remarks**

Larger test cases may also contain prerequisite states or steps, and descriptions.[8]

A written test case should also contain a place for the actual result.

These steps can be stored in a word processor document, spreadsheet, database or other common repository.

In a database system, you may also be able to see past test results and who generated the results and the system configuration used to generate those results. These past results would usually be stored in a separate table.

Test suites often also contain[9]

- Test summary
- Configuration

Besides a description of the functionality to be tested, and the preparation required to ensure that the test can be conducted, the most time-consuming part in the test case is creating the tests and modifying them when the system changes.

Under special circumstances, there could be a need to run the test, produce results, and then a team of experts would evaluate if the results can be considered as a pass. This happens often on new products' performance number determination. The first test is taken as the base line for subsequent test and product release cycles.

Acceptance tests, which use a variation of a written test case, are commonly performed by a group of end-users or clients of the system to ensure the developed system meets the requirements specified or the contract.[10][11] User acceptance tests are differentiated by the inclusion of happy path or positive test cases to the almost complete exclusion of negative test cases.[12]

See also

- Classification Tree Method

References

1. *Systems and software engineering – Vocabulary*. *Iso/Iec/IEEE 24765:2010(E)*. 2010-12-01. pp. 1–418. doi:10.1109/IEEESTD.2010.5733835 (https://doi.org/10.1109%2FIEEESTD.2010.5733835). ISBN 978-0-7381-6205-8.
2. Kaner, Cem (May 2003). "What Is a Good Test Case?" (http://www.kaner.com/pdfs/GoodTest.pdf) (PDF). *STAR East*: 2.
3. "Writing Test Rules to Verify Stakeholder Requirements" (https://www.stickyminds.com/article/writing-test-rules-verify-stakeholder-requirements). *StickyMinds*.
4. Beizer, Boris (May 22, 1995). *Black Box Testing* (https://archive.org/details/blackboxtestingt00beiz_0). New York: Wiley. p. 3 (https://archive.org/details/blackboxtestingt00beiz_0/page/3). ISBN 9780471120940.
5. "An Introduction to Scenario Testing" (http://www.kaner.com/pdfs/ScenarioIntroVer4.pdf) (PDF). Cem Kaner. Retrieved 2009-05-07.
6. Crispin, Lisa; Gregory, Janet (2009). *Agile Testing: A Practical Guide for Testers and Agile Teams* (https://archive.org/details/agiletestingprac00cris). Addison-Wesley. pp. 192 (https://archive.org/details/agiletestingprac00cris/page/n231)–5. ISBN 978-81-317-3068-3.
7. https://www.softwaretestingstandard.org/part3.php *ISO/IEC/IEEE 29119-4:2019*, "Part 4: Test techniques"
8. Liu, Juan (2014). "Studies of the Software Test Processes Based on GUI" (https://books.google.com/books?id=xK0tAwAAQBAJ&pg=PA116). *2014 International Conference on Computer, Network*: 113–121. doi:10.1109/CSCI.2014.104 (https://doi.org/10.1109%2FCSCI.2014.104). ISBN 9781605951676. S2CID 15204091 (https://api.semanticscholar.org/CorpusID:15204091). Retrieved 2019-10-22.
9. Kaner, Cem; Falk, Jack; Nguyen, Hung Q. (1993). *Testing Computer Software* (https://archive.org/details/testingcomputers00kanerich) (2nd ed.). Boston: Thomson Computer Press. p. 123–4 (https://archive.org/details/testingcomputers00kanerich/page/123). ISBN 1-85032-847-1.
10. Goethem, Brian Hambling, Pauline van (2013). *User acceptance testing : a step-by-step guide*. BCS Learning & Development Limited. ISBN 9781780171678.
11. Black, Rex (August 2009). *Managing the Testing Process: Practical Tools and Techniques for Managing Hardware and Software Testing* (https://archive.org/details/managingtestingp00rexb). Hoboken, NJ: Wiley. ISBN 978-0-470-40415-7.
12. Cimperman, Rob (2006). *UAT Defined: A Guide to Practical User Acceptance Testing*. Pearson Education. pp. Chapter 2. ISBN 9780132702621.

External links

- Writing Software Security Test Cases - Putting security test cases into your test plan (http://www.qasec.com/2007/01/writing-software-security-test-cases.html) by Robert Auger
- Software Test Case Engineering (http://www.stickyminds.com/s.asp?F=S15689_ART_2) By Ajay Bhagwat

---