Software Testdokumentation nach den IEEE und ISO Standards

Harry Sneed (TU Dresden) & Richard Seidl (Test Consultant)

Abstract: *Test documentation is needed not only to assess the extent and quality of a test but also to assess the productivity of testers and to tie the test to the software architecture. Decision makers and auditors need test documentation to decide whether to release a version or not. Maintainers need test documentation to see how changes can be tested. Testers need test documents to help them identify the causes of errors. Finally auditors require a documentation of the test to confirm that the product has been adequately tested. Test documents are prescribed by prevailing IEEE and ISO standards. Originally they were prepared by hand. Today they are most often generated automatically as a byproduct of the test automation.*

The demand for test documentation goes back to the early days of testing when the US military was testing the ballistic missile defense system. The tool RXVP – Research and Evaluation Package – instrumented the Fortran code to trace execution paths through the system and to measure the degree of test coverage. Branch coverage was considered at that time to be an important criteria for determining how well the software was tested [Miller, 1979]. It was also important to register the paths that were taken through the network of code modules in order to be able to reconstruct the tests. Shortly thereafter the IEEE published the Standard ANSI/IEEE-829 for Software Test Documentation. The first version appeared in 1983. It was later revised in 1998 [ANSI/IEEE, 1998]. It is interesting to note that this first general test standard was entitled test documentation and not just system testing. It emphasized what should come out of a system test – the results. Later in 1987 another corresponding standard for unit testing – ANSI/IEEE Std. 1008 – was published. That standard put much less emphasis on documentation, i.e. what should be delivered, and instead emphasized the process, i.e. how software units should be tested [ANSI/IEEE, 1993].

1. First Test Documentation Standard – ANSI/IEEE 829

The first test documentation standard prescribed eight test documents:

1) test plan
2) test design specification
3) test case specification
4) test procedure specification
5) test-item transmittal report
6) test log
7) test incident report
8) test summary report.

1.1 Test Plan

The test plan listed out 16 topics to be addressed in planning a test including the features to be tested, the approach to be taken, the pass/fail criteria, the suspension criteria, the test deliverables, the individual testing tasks, the test environment and staffing requirements.

1.2 Test Design Specification

The test design specification outlined the overall architecture of the test process and the tools used. It also described the features of the target system and how they were to be tested, e.g. by data comparison tools or by visual inspection. The pass/fail criteria for each feature were also given here.

1.3 Test Case Specification

The test case specification specified the purpose of each test case as well as its data inputs and outputs, i.e. pre- and post-conditions. The goal here was to communicate to the users what is being tested and to document for the testers what they should test. The test cases related the data to the functions of a system.

1.4 Test Procedure Specification

The test procedure specification prescribes the steps for executing a particular subset of test cases, or in other words, the steps used to analyze a software item in order to be able to evaluate a set of features. Here the order of the test cases and their relation to one another was defined. In today's terms this would be the test script. The test procedure is like the test cases an essential part of the test documentation making it possible to reconstruct and repeat the test.

1.5 Test Item Transmittal Report

The test item transmittal report listed out all the test items submitted for testing, including their version or revision level. Transmitted items included not only software components, but also databases, files and documents. Their location and status as well as their owners were indicated here.

1.6 Test Log

The test log was intended to provide a chronological record of all steps taken in performing the test. Each step was described together with how it was executed, when it was executed, who executed it, and what the result was. The test cases run were recorded as having passed or failed.

1.7 Test Incident Report

The test incident report is a detailed description of all anomalous events – all that happens what should not happen and all that does not happen which should. The circumstances which lead up to the event and the environment in which the event occurred are recorded. In essence the test incident report amounts to a summary of all recorded incidents which may be due to defects in the software, in the environment or in the test itself. It is the basis for the problem reports.

1.8 Test Summary Report

The test summary report summarized the results of the designated test activities and provided information necessary to evaluate the test. All variances from expected behavior were noted here together with a description of their effects. The testers were expected to give an evaluation of the system under test here, i.e. their recommendation of whether to release it or not.

1.9 Test Documentation Automation

Later on tools were developed for supporting this form of test documentation. One of the first tools to be published was TestPlan developed by the first author and Dr. Günter Rothhard in 1993. This tool was based on standard templates which were displayed to the user to be overwritten. In this way the user could adjust the model plan to his particular project. This tool became very popular in the 1990's [Sneed/Erdoes, 1995].

2 Current State of Test Documentation ISO/IEC-29119

Over the years the IEEE Standard has gone through several revisions but the basic structure has remained the same. We are still concerned with producing a minimum of test documentation to state what we intend to test, how we intend to test it and what we have tested. There are still test cases and test procedures, test logs and anomaly reports, albeit in another form and in an automated environment.

The latest test documentation standard is to be found in chapter 3 of the new standard ISO/IEC 29119 [ISO/IEC/IEEE, 2013]. This standard defines the use and content of software test documentation used throughout the defined multi-layer test process. Test documentation is identified for the three layers of the test process are:

- Organizational Test Process;
- Test Management Processes;
- Static Test Processes and Dynamic Test Processes.

2.1 Documentation of the organizational Test Process

The documents developed in the organizational test process comprise the following types:

- Organizational Test Policy;
- Organizational Test Strategy.

The **organizational test policy** is a document describing company-wide rules and processes for testing software systems in general. This document should exist once within each organization and is valid for all projects carried out within that organization. It lays down the policy to be followed, for instance if testing is to be done in a separate team or whether testers are to be integrated in the development teams or when testing is to be outsourced to a third party.

The **organizational test strategy** identifies the roles and test team composition. The information may be presented in a table or use a diagram to show test organization hierarchy. It prescribes what test documents have to be delivered and which may be delivered. It regulates among other things, the test exit criteria which used to be defined for each and every project, the degree of independence of the test team, what test documents are to be delivered, which test techniques should be used, what test tools are to be used and how incidents are to be handled. In this respect, it is closely tied to the general test processes and describes the test phases through which the test process must pass. As such it serves as a general framework for all test projects within the organization.

2.2 Documentation of the Test Management Processes

The documents developed in the test management processes are of the following three types:
- Test Plan;
- Test Status Report;
- Test Completion Report.

The **Test Plan**, produced for each test project, is similar to the one that existed before. It refers to other documents which need to be considered such as the requirement document, the prevailing standards, the design model, the user guide and operation guide. It lists out the test items – the components, the user interfaces, the system interfaces and the databases, that have to be tested. It sets the test scope by by specifying what functionality is being included (inclusions) and equally important, what functionality is being excluded (exclusions) from the test effort. The test plan identifies the dependencies on other products as well as on other processes. Of particular importance is the risk analysis, e.g. hazard analysis, of possible system failures which have to be given more attention in testing. The new test plan also defines the test completion criteria, the test environment, the test data requirements, the test tool requirements and project specific test deliverables. As opposed to earlier, the new test plan includes a test cost estimation. Project specific staffing and training requirements are identified and defined. In the end the test plan should be approved by the project stake holders.

The **Test Status Report** has been added**.** It is a report on the current status of the test project, what has been tested so far and to what degree. It is produced in connection with each new release. This could be as often as every two weeks. It documents test progress, i.e. how many of the planned test cases have really been tested, and how high is the test coverage. It identifies the test bottle necks and proposes ways of circumventing them. It uses metrics like defect density, defect rate, response times, throughput rate, test case coverage and code coverage to exactly record the test status. In agile testing the test status report is the most significant document stake holders will receive. It tells them where the project is relative to where it should be.

The **Test Completion Report** corresponds to the old test summary report. It summarizes the testing performed in the different test phases and for the different test types for the project as a whole. Only the applicable testing is included. if it is a test completion report for an individual test phase or test type. It provides details on what was tested, and describes any constraints on how the testing was performed, e.g., restrictions on test environment availability. The test completion report should include the final test coverage report and the defect statistics. Based on the final completion report managers should be able to decide whether to release the product or not.

2.3 Documentation of the static and dynamic test processes

The documentation of the static and dynamic test processes goes into detail in how to conduct the test. There are two ways of testing, one static and one dynamic.

2.3.1 **Static test documentation** is focused on analyzing code and documents. It can be performed either manually or dynamically. If done manually, a person reads through the code or documents and locates deficiencies which may result in defects, deficiencies such as failing to check input parameters or output results. The documentation is written by the person reviewing the document or inspecting the code. If done automatically, a tool scans through the documents or the source code to recognize such deficiencies. The reports listed below are then generated automatically by the tool. The documents produced in the static test process comprise the following types:

- Review Checklist
- Static Analysis Rules
- Incident Report
- Action Item List
- Static Test Report.

2.3.2 **Dynamic test documentation** is focused on executing the system under test. It too can be performed either manually or automatically. In either case the following reports are recommended by the standard:

- Test Specification; divided into
  - Test Design Specification
  - Test Case Specification
  - Test Procedure Specification
- Test Data Requirements
- Test Environment Requirements
- Test Environment Readiness Report
- Test Outcome
- Test Results

- Test Execution Log
- Incident Report

2.4 Comparison of the new standard with the old

In this respect the new standard is similar to the old one. New are the test environment readiness report and the test outcome. Also the test case specification has been extended to include coverage information. Different types of test coverage are included such as equivalence class coverage and use case coverage in addition to code coverage. The new standard puts particular emphasis on the documentation of test data requirements. The purpose of the new Environment Readiness Report is to describe the status of each environment required. A dynamic test may require many different environments. The test outcome document is a registration of the actual results provided by the test item as a result of the execution of a test case. This is what must be compared to the expected results to validate the test result.

We can summarize that the new standard ISO/IEC 29119 requires a much broader set of test documents and goes into greater detail as to the content of those documents. It also provides recommendations on how to prepare the documents. Not only that, it even offers the users templates to help structure the documents. Organizations producing safety and mission critical software should certainly strive to provide all of the documents required by the new standard. Other organizations producing less critical software should at least make a subset of the documents. Which subset that is will depend on the test process they are following. In any case test documentation is essential and should be taken seriously.

3 Current Test Documentation Practice

In today's practice the degree of test documentation depends on the size and type of the project. In large projects more test documentation is produced than in smaller ones. In conventional development projects with a phase model more documentation is produced than in agile projects which strive for a minimum of test documentation unless the documentation is automated. A test plan is usually made even in agile projects where there is a test plan for each release. It is referred to as the one day test plan [Crispin/Gregory 2009]. It outlines the deliverables, the deadlines and the test goals. In conventional projects test planning may take up to two weeks to produce a plan similar to the one specified in ANI-IEEE 829. A test design is usually only made in larger projects where the system has a complex architecture. Testers are not sure what should go into the design so they tend to reduce it to a single diagram, often a context diagram. The new standard can help here. Test cases are usually specified in an excel type table. If users are using an automated test management system such as IBM's Rational Tester or HP's Test Center they will store the test cases in the tool database where they can better process them. Test procedures may be documented with text files or, if they are using an automated tool to perform the test, they will be documented in the form of a test script. Many users have tools to document code coverage, especially at the unit test level, but few users document the test outcome as required by the new standard. What is usually well documented is the software performance. What is less documented is the functional correctness.

Where testing is done by a separate testing team, test status reports are prepared on a regular basis and a final test summary report is written to summarize the costs and benefits of the test, i.e. how many defects were reported and how many hours were tested. This report is seen by management as a justification of the test costs. Where testers are integrated together with developers as in agile development, there is less formal test documentation and what is produced is most often produced by tools. Reported defects are posted on the project white board while verbal reports are made by testers at the project meetings [Linz, 2012]. In agile development emphasis is placed on automated testing where test documentation is mainly a byproduct of the individual test steps – design, test case specification, test scripting, test execution and test coverage evaluation [Polo/Reales/Plattini,2013].

References:

[ANSI/IEEE, 1993] ANSI/IEEE Standard 1008 – 1993: Software Unit Test Standard, Institute of Electronical and Electronic Engineers, New York, N.Y. 1993.
[ANSI/IEEE, 1998] ANSI/IEEE Standard 829 – 1998: Software Test Documentation, Institute of Electronical and Electronic Engineers, New York, N.Y. 1998.
[Crispin/Gregory, 2009] Crispin, L, Gregory, J.: Agile Testing – A Practical Guide for testers and Agile Teams, Addison-Wesley, Upper Saddle River, NJ, 2009
[ISO/IEC/IEEE, 2013] ISO/IEC-29119-3-2013: Software Test Documentation, International Standards Organization, Geneva, 2013
[Linz, 2012] Linz, T.: Testing in Scrum Projects, dpunkt Verlag, Heidelberg, 2012 .
[Miller, 1979] Miller, E.F.: "Software Testing and Test Documentation", IEEE Computer Magazine, March, 1979, p. 56:
[Polo/Reales/Plattini,2013] Polo, M., Reales, P. Plattini, M.: "Test Automation" IEEE Software Magazine, Jan. 2013, p. 85
[Sneed/Erdoes, 1995] Sneed, H./Erdoes, K.: "Automating the ANSI/IEEE Standard 829 for Test Documentation", Proc of 2nd IEEE Conference on Software Standards, Brighton, G.B., Sept., 1995, p. 101