Frank Witte

Strategy, Planning and Organization of Test Processes

Basis for Successful Project Execution in Software Testing

Springer

Strategy, Planning and Organization of Test Processes

Frank Witte

Strategy, Planning and Organization of Test Processes

Basis for Successful Project Execution in Software Testing

Springer

Frank Witte
Landshut, Germany

ISBN 978-3-658-36980-4 ISBN 978-3-658-36981-1 (eBook)
https://doi.org/10.1007/978-3-658-36981-1

This Springer imprint is published by the registered company Springer Fachmedien Wiesbaden GmbH, part of Springer Nature.
The registered company address is: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

Foreword

With my first book *Softwaretest und Testmanagement* (*Software Test and Test Management*), I dealt with an overview of the entire testing process, and with my book *Metriken für das Testreporting* (*Metrics of Test Reporting*), I presented analyses and suitable procedures that are used during and at the end of the testing process for control and monitoring. With the present book, I would like to deal more with the beginning of the test process, I mainly describe a test phase where a test object does not necessarily exist yet and that evolves around test strategy, test conception, and planning. The organization and planning of projects basically extend over the entire project period, as it must be constantly adapted to new circumstances. However, one focus is on the concept phase at the beginning of the project.

Back in the day, there were still companies that seriously questioned the need for software system testing. After all, that has changed due to awareness across the board, whether it is a small business or a corporation. Every professional software developer learns in college how to test their software. Depending on whether it is a first release or an update, it is important to test the software as a whole or as part of regression testing as long as it makes sense. Not doing so is similar to a surgeon discussing whether disinfecting his hands before surgery is really necessary.

In project management, the rule is that 25–35% of the expected total project duration should be spent on planning. The clearer the objectives, framework conditions, and requirements are described, the easier and faster the project can be implemented later. Just as project planning is the main task of project management, well-thought-out and sound test planning is also key to a successful test project.

This book deals with strategy, planning, and organization of test processes.

Strategy was originally understood as the art of army command. Clausewitz further developed the military concept politically and economically. In business, the planned behaviors of companies to achieve their goals are understood as strategy. Consequently, a test strategy defines framework conditions and procedures but also necessary individual activities for a project, which are required for the successful implementation of the goal.

Planning is essential to shape the future and has to consider which means can be used to achieve which goal. For software testing, this means that goals, resources, and processes must be defined and documented. Thanks to planning, processes are constantly

determined that are run in a structured manner in the operational environment. The results of planning are constantly implemented in the organization. This means that test processes must be continuously controlled, that is, the current test progress must be permanently compared with the planning, and corrective measures must be initiated if necessary.

A test concept is created for planning test activities in a software project. The test concept can be compared to a construction plan. The structure of the book is therefore based on the structure of a test concept according to the IEE829 standard. In addition, I provide guidance on test strategy, project and test planning, and methods for evaluation. New trends such as agile or cognitive testing also have an impact on the test organization.

The definition "IEEE 829 Standard for Software Test Documentation" is a standard published by the IEEE (Institute of Electrical and Electronic Engineers), which describes a set of eight basic documents for the documentation of software tests. The current version is IEEE 829-2008. The structure according to IEEE 829 can be adapted and extended, and not all parts are relevant to the same extent. However, it is useful to use it as a rough guideline and compass.

The test concept determines the delimitation, procedure, and means used and contains a schedule of the test activities. The test concept identifies the test objects, the features to be tested, and the test tasks. It forms the basis on which the test organization and the test infrastructure are provided, and the tests are performed.

In earlier times, testing activities usually started at a later stage of the project. This meant that the test department was faced with scheduling conflicts right from the start. Meanwhile, it has become widely accepted that the test manager should be involved in the project from the very beginning, that is, they should accompany the development process from the very beginning, and that the planning phase should therefore take place at the start of the project. Consequently, the test concept should also be created before or while the system development is in progress and the project phase of defining suitable test cases and the test execution phase have not yet begun.

The same applies to test planning as to the overall planning of projects: for efficient and effective project planning, a "rolling" approach is recommended. This means that at the beginning of the project, a rough plan is made for the entire project, which is then outlined in detail for each individual upcoming project phase. The advantages of this approach are that knowledge gained during the investigation can be incorporated into the planning and the planning effort remains reasonable [ORGH2019].

In the following, however, I would not only like to show the contents of the test concept like from a textbook but also point out the pitfalls and peculiarities in practice from experience. Already during the development of the test concept one can, with practiced observation, recognize risks arising during the project in advance and define possible countermeasures from the beginning. You can point out the weaknesses in the process and make the framework conditions transparent to be prepared from the outset if problems arise. The point is not to be a driven person in reaction mode as a test manager but to be able to act actively right from the start.

Every test project is embedded in a particular environment, and I recommend that every test manager first understand the context and the particular issues involved:

On the one hand, there is the issue of personnel: Are there enough testers available at all? Are the testers basically motivated? Are certain resources overloaded from experience? How is the workload over time? If an employee is planned with a capacity of 40 working days from 1 March to 30 April, but his work package has prerequisites that are not available until 15 March and the employee already has to deliver his test results on 15 April, he is massively overloaded at times, although the overall availability is correct. Are there (possibly from the past) personal differences in the team or coalitions? Are the employees exclusively available for the project, or do different projects access the individual testers? It is important to consider that each stakeholder makes communication more complicated: With only 10 stakeholders, there are 45 communication channels between all stakeholders, with 20 there are already 190. Especially, the "grapevine" can destroy a lot in larger organizational units.

Who will conduct the test, are the testers experienced or new to the project? Are they motivated or rather disillusioned, or do they perhaps even make a bad mood against the project? Are there personal differences within the stakeholders or alliances in the company, for example, developers versus testers?

What is the status of the project, is it politically desired, is it a high priority? Does the project have supporters in the company, or is it possible that it is being slowed down, blocked, or put on the back burner due to other priorities? Are there other projects with a higher priority, are a prerequisite for the upcoming project, or perhaps even contradict it?

Are there technical risks, innovations, or unknown side effects? A change of servers or data storage in the cloud is usually not directly related to the software under test, but in many cases, there are effects. A change in databases, for example, is similar. Are there other projects running in parallel that have a technical or business impact on your own project, or is the project under consideration even at their mercy? Which requirements have to be met to start certain activities at all or to be able to carry them out successfully? Do other projects depend on your own project?

How did projects work in the past? Are there company-specific or industry-specific problems? Are there "lessons learned protocols" for this?

A test manager who comes into a project from the outside (e.g., externally) often does not know enough about the technical and specialist features and the strengths and weaknesses of the individual employees. Someone who has been with the company for a long time may know all this, but again may be biased because of experiences from previous projects. What is the position of the test department in the company? Is it strong, or is testing rather considered as a necessary evil that ekes out a shadowy existence?

In the past, application testing was often seen as a necessary but not very useful appendage of classic software development projects. This view has changed, not least against the background that testing, considered over the lifecycle of an application, takes up between 25 and 50% of the total costs incurred. Software quality problems can also have serious

consequences, such as a quality-related project delay, which often extends the time-to-market. Major commercial and material damage can occur if, for example, a booking platform fails for software-related reasons, important data is lost or falsified, or security gaps in the system lead to the theft of customer or credit card information. In addition, the ever-closer integration of business and IT increases the demands on time-to-market and quality, which increases the pressure on the test organizations to perform. At the same time, systems and IT landscapes are becoming increasingly complex, and software development and life cycles are becoming faster. In addition, regulatory and security requirements are becoming more stringent. In this respect, it is only logical that application testing has been one of the current focus topics of IT for several years [IODT2019].

From my point of view, unclear task definitions are a major concern in numerous test projects. Many of the communication problems are caused by the fact that the test object is not exactly defined from the start, how it is to be delimited, and what the expectations of the software test are. Experience has shown that requirements which are incomplete, interpretable, or contain contradictory content are unfortunately the rule rather than the exception. These problems then lead to defects during development and queries and bugs in the test process.

What expectations does management have of the project? Are these expectations realistic in terms of deadlines, costs, and quality? Where is the responsibility for costs located? Especially the test manager, who is supposed to ensure the quality of the software, has to warn against overly optimistic expectations right from the start.

Only when the corresponding statements are backed up with figures do they become concretely tangible.

For the evaluation, I recommend always converting the existing risks into real euros. A statement like “the risk is €750.000 due to insufficient test coverage” is much more forceful than a statement like “probably 5% of the planned test cases cannot be tested on time.”

Many questions can be asked in advance and considered at a suitable point in the test concept. I will name typical problem cases in the following chapters. In some cases, it is recommended to create corresponding “help documents” for this purpose.

This book is intended to help bring the testing concept to life. Sometimes, the test concept is only created for reasons of audit security. Sometimes, it exists as a theoretical document, but does not reflect the actual state of the testing activities. This wastes the opportunity to individually design the test process with the necessary pragmatism and to document it in a well-founded manner. A test concept with an alibi function does not serve its purpose.

The released version is a plan that you start with. It is normal to gain new insights during testing activities and thus no longer be able to test strictly according to the predefined path. You have to be flexible enough to proceed pragmatically, and there can always be good reasons to deviate from the original plan. But, in this case, it is also necessary to adapt the test concept afterwards so that the actual test procedure emerges from it. If a wall is added to a new house on the construction site or a door is moved, the construction plan must also be changed. The same applies to the test concept.

The more the individual team members refer to the test concept when they have unresolved questions about software testing, the more they deal with it and make critical comments, and the sooner the entire test project can be steered in the right direction. The test manager in particular has the responsibility to make sure that the test concept is changed, if necessary, when current developments overtake the plan. Often, one learns rather incidentally that certain assumptions defined in the test concept are not applicable at all.

The test concept is not a product that is created at the green table. A test manager does not sit in a quiet room like a musical genius and creates his great work. A test concept is developed step by step in dialogue with experts and those who will be affected by the project in the future. For small projects, a few working days may be enough (to be fair, it should be mentioned here that a test concept requires an average of 5 to 20 PT of effort, for particularly extensive projects (e.g., multi-projects or large software packages)) even significantly more. To develop a test concept, a large number of necessary questions must be clarified with the right experts. Of course, individual details still need to be worked out afterwards, but the most important things should be outlined and thought through together. Alternatively, a test manager, test coordinator, or test responsible (or whatever you want to call them) can clarify the questions in individual interviews and collect everything bit by bit [PERF2017].

In agile teams, where work is iterative, not all questions have to be answered in advance. It is also sufficient to address them and to tackle their solutions in order of priority. For example, it is hard to see why a team has to define exactly which test artifacts (test cases, test protocols, reports, ...) it wants to keep from the beginning. But if an acceptance test by the business department is required for the next release, it is better to book the target group already now (and thus inevitably commit oneself). The above standard in the form of a task list can visualize the necessary clarifications well and ensure that you do not solve or decide anything too late during iterations. Increasing agilization and future trends lead to new environmental conditions to which the test concept must also adapt.

A central point applies to all elaborations and determinations: You need the unconditional agreement of those involved! Only if there is a consensus in the company, it is at all possible to achieve the goals of quality assurance which are implemented thanks to tests.

In this context, it is particularly important to subject the test concept to a review and to define the review process precisely. Review meetings with all participants are essential for this. Simply sending a test concept around and waiting for feedback is not effective. Not everyone involved can say a few words on all points in the test concept, but if you only ask for feedback, you run the risk that individual pages are not even looked at because an invited reviewer for certain chapters considers someone else to be responsible for assessing the content. Paper doesn't blush. Especially with reviews, I have often observed that people think they don't have time and therefore want to shorten or refrain from necessary activities. In the end, however, they use up even more time – according to the motto "we have time to do things wrong three times but not to do them right once."

A review cycle is therefore not used to find defects in the concept alone but, above all, to obtain approval. You have to actively seek this approval. Many people make the mistake

of slapping a concept on the table and then assuming that everyone will (obligingly) observe it. Systemically, the responsibility lies with the developer of the test concept, and they will have to take care of everything themselves, which rarely makes sense [PERF2017].

Every project should have its own test concept, depending on the scope of the project.

A project is defined by the DIN 69901 standard as follows: It is a project that is essentially characterized by the uniqueness of the conditions in their entirety – there is a concrete target for the project, it is limited in time, it requires specific resources (e.g. financial and personnel), there is an independent process organization that is distinct from the standard organization of the company, non-routine tasks are handled, it consists of at least three participants, and has a minimum duration of 4 weeks.

A project has a beginning and an end, unlike a process, which is continuous. A project structure plan (PDF) is divided into work packages in which the individual activities, that is, steps to be carried out, are described. Often, a milestone is set at the end of a work package – a point in time at which the progress of the project is analyzed and, if necessary, presented to the clients in a meeting. The project manager is responsible for monitoring the process, and all members of the project team must work with the project manager and are responsible for the tasks assigned to them. Crisis management and the creation of documentation are also part of the management tasks [KARB2019].

The test manager is responsible for the project management of all test levels, and they coordinate and monitor the test activities. A good cooperation and close regular coordination with the project manager should therefore be given, otherwise it will be difficult or impossible to achieve good progress and results in the test process. Differences with the project manager should not arise in the first place or must be resolved immediately. If the project manager and the test manager do not pull in the same direction, other stakeholders will quickly notice this and the productivity of the project will suffer, or even the entire project will fail.

However, it is also possible to create a generic test concept for the entire company or the entire organizational unit under consideration, especially if recurring activities are involved.

The IEEE 829 serves as a guideline, but it is necessary to adapt the test concept to the operational conditions and individual characteristics. I have already created a number of test concepts, which are similar in the rough structure but differ in focus depending on the industry, company, and test object.

Experience shows that there are numerous questions that can be clarified during the creation of the test concept. The more stakeholders are involved in the planning phase, the more attention and trust are generated in advance, before the actual test implementation begins.

With official specifications and standards, one usually assumes an optimal working environment. This is often not the case. Instead, from the very beginning, one has general conditions that negatively influence the project, such as unclear tasks, delayed deadlines, and a lack of know-how in the test team: communication problems are usually responsible for this. The examples in many textbooks assume ideal environmental conditions that do

not exist in practice. In the following, I will go into detail about how software testing can still be successful or how the worst can be prevented, especially under difficult boundary conditions. Many projects do not reach their goal in time and budget, often because planning was too optimistic from the beginning and/or too few parameters were considered. However, it is possible to use suitable methods to ensure that delays, cost overruns, or quality deficiencies are at least significantly minimized. To achieve this, one must walk along the royal path between a planned, structured procedure and the necessary flexibility. A pragmatic approach to existing realities is just as important as compliance with rules and standards.

The scope of a test concept certainly depends on the size of the project. But even for small projects, you need to have a plan for how to proceed with testing and document the crucial framework data.

If information already exists or is maintained elsewhere, redundant descriptions should be avoided. A stakeholder list with names, department designations, and telephone numbers must be updated in the event of an operational reorganization. However, describing only the role and name and referencing a document that is maintained for it anyway can be better, especially for large projects. This also applies to schedules. Redundant data maintenance always involves the risk that different, and therefore contradictory, information is kept, on which to build.

It is recommended to prepare a template for the test concept for an organizational unit, that is, to perform tests more frequently. If individual parts should not be relevant, this can then be noted in a suitable point in this template. This way, nothing will be forgotten, and you will already get hints on what to think about when creating the test concept.

You can orientate yourself on similar projects from the past and examine weak points that occurred at that time already at the beginning of the project: if you already know that the provision of the test environment was delayed several times in earlier releases or if you already suspect that there will be repeated defects during the integration of certain components, you can already plan sufficient buffers during the planning phase and point out corresponding risks already at the beginning of the project.

It should be borne in mind that a good test organization and a well-thought-out test concept require time. The estimates for the effort required for this should be based on experience, project scope, and the purpose of the application. Here, it is important for the test manager to stand behind his figures, to give these activities the necessary importance in the company, and to put the brakes on some excessive optimism.

Landshut, Germany Frank Witte

References

[KARB2019]: https://karrierebibel.de/projektarbeit, accessed on 9 May 2020
[PERF2017]: https://www.informatik-aktuell.de/entwicklung/methoden/das-perfekte-testkonzept-in-6-schritten.html, accessed on 9 May 2020
[IODT2019]: https://www.iot-design.de/allgemein/transformation-von-testorganisationen/, accessed on 9 May 2020
[ORGH2019]: https://www.orghandbuch.de/OHB/DE/Organisationshandbuch/2_Vorgehensmodell/21_Projektvorbereitung/212_Projektplanung/projektplanung-node.html, accessed on 9 May 2020

Contents

List of Figures

List of Tables

Test Documents According to IEEE 829

1

Abstract

IEEE 829 defines basic documents for software testing. The test concept describes the results of the test planning. Different types of test documents document the test contents, the test progress and the results of the test execution.

1.1 Standards

Standards help standardize business processes and procedures. A standard is a document that specifies requirements for products, services or processes. It thus creates clarity about their properties, facilitates the free movement of goods and promotes exports. It supports rationalization and quality assurance in business, technology, science and administration. Furthermore, it serves the safety of people and property as well as the improvement of quality in all areas of life [DIN2019].

The standardization process in software testing started much later than in other industries. For example, the certification process in the field of software testers was not started until the mid-1990s in Great Britain, which gave rise to the ISEB certification program in 1997 (ISEB = "Information System Examination Board").

In 1998, a first variant of IEEE 829 was described to establish certain test documents at all. This variant was expanded and updated in 2008.

The definition **IEEE 829 Standard for Software Test Documentation** is a standard published by the IEEE (Institute of Electrical and Electronics Engineers) that describes a

F. Witte, *Strategy, Planning and Organization of Test Processes*,
https://doi.org/10.1007/978-3-658-36981-1_1

set of eight basic documents for the documentation of software tests. The current version is IEEE 829–2008, which describes the form and content of each document. However, it does not prescribe which of the respective documents must be used.

This standard can be used for traditional, waterfall model software projects as well as for agile projects.

In September 2013, the first parts of the international standard **ISO/IEC/IEEE 29119** were published, replacing the IEEE 829 Standard for Software Test Documentation internationally.

The standard describes eight documents, which can be divided into three categories, as follows.

1.2 Test Concept

Test Plan: The test plan defines the scope, approach, means and schedule of the test activities. It determines the elements and product functions to be tested, the test tasks to be performed, the personnel responsible for each task, and the risk associated with the concept.

The test concept serves as a guideline for planning and organizing test activities.

The following shows a structure that has proved to be useful for creating test concepts.

1. Introduction
   - 1.1. Identification of the test concept
   - 1.2. Scope and coverage
   - 1.3. References
     - 1.3.1. External references
     - 1.3.2. Internal references
   - 1.4 System to be tested and test object
   - 1.5. Overview of the test items
     - 1.5.1. Organization
     - 1.5.2. Project test plan
     - 1.5.3. Integration stages
     - 1.5.4. Resource overview
     - 1.5.5. Responsibilities
     - 1.5.6. Tools, techniques, methods, metrics
2. Details
   - 2.1 Test process and test levels
   - 2.2. Documents
   - 2.3. Deviation and change management
   - 2.4. Reporting

3. General
   3.1. Glossary
   3.2. Change service and history [QYTE2019].

The test concept defines the scope, procedure, scheduling and required resources of the intended test activities. Test objects, features to be tested and the individual test tasks are identified.

Test tasks can also be assigned to individual employees. Sometimes, reference is made to a separate planning document for this purpose.

The test should take place independently of the development. Only if the test is a neutral, separate entity it will be able to guarantee this independence. The resulting organizational definition is also described in the test concept.

In addition, the test environment, the test design procedure and the procedures to be used for measuring the test activities and test results are described, and their selection is justified. The test concept includes a chapter on risks and possible countermeasures if these risks occur.

The test concept thus documents the results of the test planning process.

1.3 Test Specification

The test specification contains the documents, i.e., the test design specification, test case specification and test procedure specification.

The test design specification refines the description of the procedure for testing the software. It identifies features and functions that need to be covered by defined appropriately assigned logical test cases. It also describes the test cases and test procedures that are required for the test cases to be considered successfully completed. The criteria according to which tests are passed or failed are defined for each function [SETH2019].

The test case specification defines the input values to be used and the expected output values. In addition, the test case specification describes the necessary pre- and post-conditions, the objectives and test actions. The priority and duration of the test execution should also be documented, especially for test cases to be performed manually. If necessary, it should also be described how the system must be reset to an initial state after the test execution.

In the IEEE 829 standard, test cases are separated from the test design. This allows the test cases to be used in different designs and reused in other situations. In practice, the test design specification and test case specification are often stored together in one document and existing test cases are only copied for a new project. However, the modular structure

in different documents leads to a higher quality, even if it is an additional effort at the beginning.

The test procedure specification is the description of all steps for the execution of the specified test cases and the implementation of the associated test design. For many test cases, the prerequisites must first be established (e.g., several booking steps) before the actual test case can even be started. Some tests consist of a series of steps. The test procedure specification is sometimes also called test procedure description, test script or test script. Mixed forms of test case specification and test procedure specification are also often used.

In many professional test management tools, the test procedure and the separation of test case and test procedure specification are mapped through different areas.

1.4 Test Reporting

The reporting system should summarize the results of all test runs of a version.

The test item transmittal report describes the transmission of test cases in case separate development and test teams are involved, or in case an official time for the start of a test execution is desired. In the test item transmittal report, a defined version is handed over to a test team, and the testable features are listed there.

The test log is used to chronologically record the events during a test execution. For each test execution, it must be defined which log files are required by the development to be able to reproduce misbehavior. This is especially important for automated tests that run overnight or on weekends. However, it should be noted that a high log level not only records a lot of information and therefore requires a lot of memory, but also that the increased logging can increase the response time and the system load to such an extent that defects occur that would not occur under normal operation.

The test incident report describes all events that occur during test execution and require further investigation. According to IEEE 108 (IEEE Standard for Software Unit Testing), a deviation is an event that occurs during testing and requires further investigation.

The test summary report summarizes the test activities related to one or more test design specifications. For example, when the system test or integration test of a defined software is complete, the test summary report is created to start the next test stage. If, for example, there is only a conditional release during the system test, the software can still be tested in the acceptance test, but certain restrictions must be observed. Sometimes, at the end of the test activities of a test level, defects have not yet been eliminated or successfully retested, so that the next higher test level can be carried out with these restrictions, but the product cannot yet be delivered in production as long as these bugs still