CSTE
Certified Software Tester

QUALITY ASSURANCE INSTITUTE
WORLDWIDE

CHAPTER NETWORK

BEST PRACTICE
An 11-Step Software Testing Process Example

The software testing process example, as illustrated in Figure 19, is an 11-step testing process that follows the “V” concept of testing. The “V” represents both the software development process and the 11-step software testing process. The first five steps use verification as the primary means to evaluate the correctness of the interim development deliverables. Validation is used to test the software in an executable mode. Results of both verification and validation should be documented. Both verification and validation will be used to test the installation of the software as well as changes to the software. The final step of the “V” process represents both the development and test team evaluating the effectiveness of testing.

Note: The terms in this example vary slightly from the SDLC example to illustrate there are no common definitions used by all IT organizations.

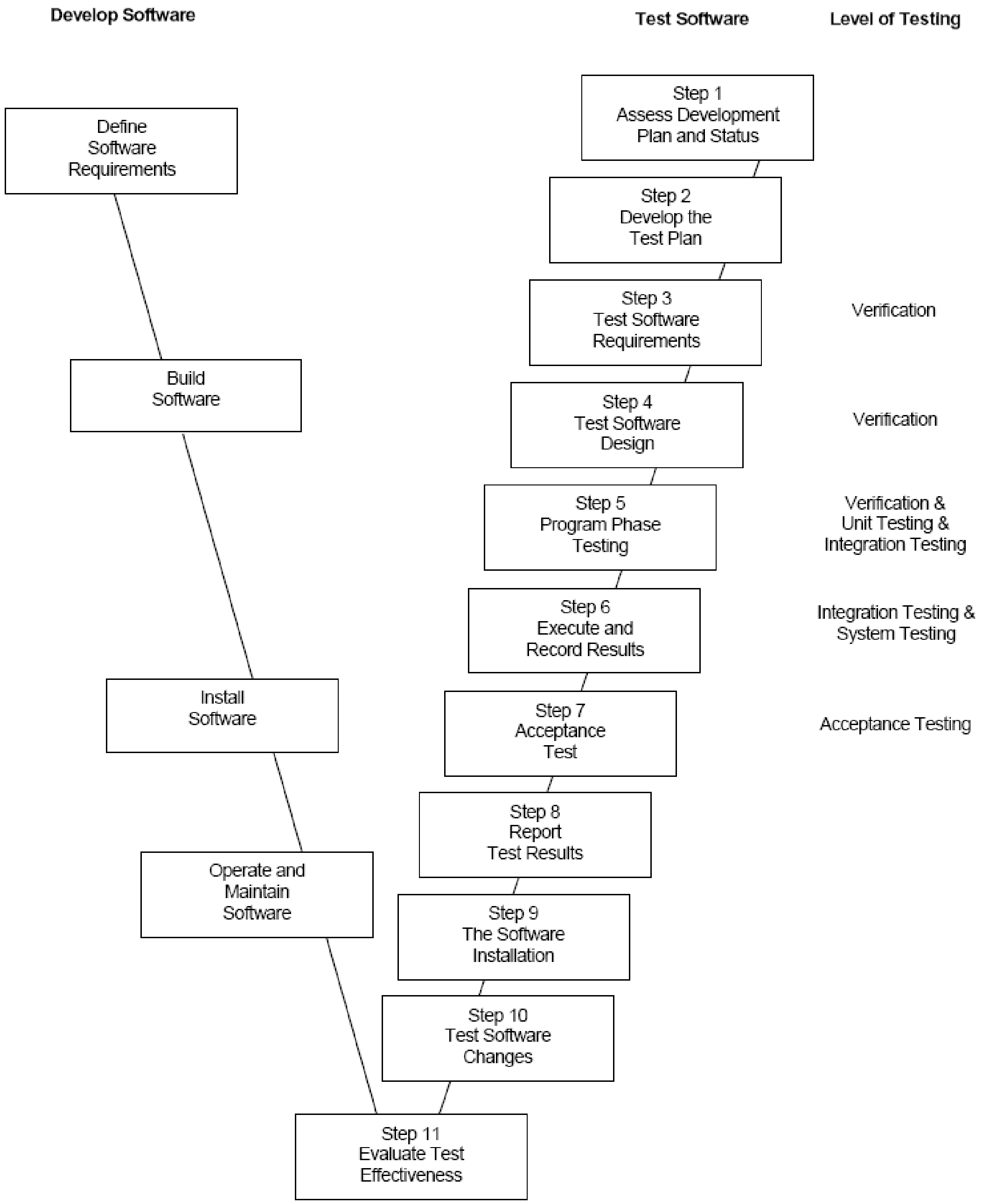
Develop Software
Test Software
Level of Testing
Define Software Requirements
Build Software
Install Software
Operate and Maintain Software
Step 1 Assess Development Plan and Status
Step 2 Develop the Test Plan
Step 3 Test Software Requirements
Verification
Step 4 Test Software Design
Verification
Step 5 Program Phase Testing
Verification & Unit Testing & Integration Testing
Step 6 Execute and Record Results
Integration Testing & System Testing
Step 7 Acceptance Test
Acceptance Testing
Step 8 Report Test Results
Step 9 The Software Installation
Step 10 Test Software Changes
Step 11 Evaluate Test Effectiveness

Figure 19. The 11-Step Software Testing Process Example

Step 1: Assess Development Plan and Status

This first step is a prerequisite to building the VV&T Plan used to evaluate the implemented software solution. During this step, testers challenge the completeness and correctness of the development plan. Based on the extensiveness and completeness of the Project Plan the testers can estimate the amount of resources they will need to test the implemented software solution.

Step 2: Develop the Test Plan

Forming the plan for testing will follow the same pattern as any software planning process. The structure of all plans should be the same, but the content will vary based on the degree of risk the testers perceive as associated with the software being developed.

Step 3: Test Software Requirements

Incomplete, inaccurate, or inconsistent requirements lead to most software failures. The inability to get requirements right during the requirements gathering phase can also increase the cost of implementation significantly. Testers, through verification, must determine that the requirements are accurate, complete, and they do not conflict with one another.

Step 4: Test Software Design

This step tests both external and internal design primarily through verification techniques. The testers are concerned that the design will achieve the objectives of the requirements, as well as the design being effective and efficient on the designated hardware.

Step 5: Program (Build) Phase Testing

The method chosen to build the software from the internal design document will determine the type and extensiveness of tests needed. As the construction becomes more automated, less testing will be required during this phase. However, if software is constructed using the waterfall process, it is subject to error and should be verified. Experience has shown that it is significantly cheaper to identify defects during the construction phase, than through dynamic testing during the test execution step.

Step 6: Execute and Record Results

This involves the testing of code in a dynamic state. The approach, methods, and tools specified in the test plan will be used to validate that the executable code in fact meets the stated software requirements, and the structural specifications of the design.

Step 7: Acceptance Test

Acceptance testing enables users to evaluate the applicability and usability of the software in performing their day-to-day job functions. This tests what the user believes the software should perform, as opposed to what the documented requirements state the software should perform.

Step 8: Report Test Results

Test reporting is a continuous process. It may be both oral and written. It is important that defects and concerns be reported to the appropriate parties as early as possible, so that corrections can be made at the lowest possible cost.

Step 9: The Software Installation

Once the test team has confirmed that the software is ready for production use, the ability to execute that software in a production environment should be tested. This tests the interface to operating software, related software, and operating procedures.

Step 10: Test Software Changes

While this is shown as Step 10, in the context of performing maintenance after the software is implemented, the concept is also applicable to changes throughout the implementation process. Whenever requirements change, the test plan must change, and the impact of that change on software systems must be tested and evaluated.

Step 11: Evaluate Test Effectiveness

Testing improvement can best be achieved by evaluating the effectiveness of testing at the end of each software test assignment. While this assessment is primarily performed by the testers, it should involve the developers, users of the software, and quality assurance professionals if the function exists in the IT organization.

References

Guide – CSTE Common Body Of Knowledge, V6.1