BEST PRACTICE
When Should Testing Occur?

The traditional view of the development life cycle places testing prior to operation and maintenance as illustrated in Table 5. All too often, testing after coding is the only verification technique used to determine the adequacy of the system. When testing is constrained to a single phase and confined to the later stages of development, severe consequences can develop. It is not unusual to hear of testing consuming 50 percent of the development budget. All errors are costly, but the later in the life cycle that the error discovered is made, the more costly the error. An error discovered in the latter parts of the life cycle must be paid for four different times. The first cost is developing the program erroneously, which may include writing the wrong specifications, coding the system wrong, and documenting the system improperly. Second, the system must be tested to detect the error. Third, the wrong specifications and coding must be removed and the proper specifications, coding, and documentation added. Fourth, the system must be retested to determine that it is now correct.

Table 5. Life Cycle Verification Activities

Life Cycle Phase	Verification Activities
Requirements	- Determine verification approach - Determine adequacy of design - Generate functional test data - Determine consistency of design with requirements
Design	- Determine adequacy of design - Generate structural and functional test data - Determine consistency with design
Program (build/construction)	- Determine adequacy of implementation - Generate structural and functional test data for programs
Test	- Test application system
Installation	- Place tested system into production
Maintenance	- Modify and retest

If lower cost and higher quality systems are the information services goals, verification must not be isolated to a single phase in the development process, but rather, incorporated into each phase of development. One of the most prevalent and costly mistakes on systems development projects today is to defer the activity of detecting and correcting problems until late in the project. A major justification for an early verification activity is that many costly errors are made before coding begins.

Studies have shown that the majority of system errors occur in the design phase. These numerous studies show that approximately two-thirds of all detected

system errors can be attributed to errors made during the design phase. This means that almost two-thirds of the errors must be specified and coded into programs before they can be detected. The recommended testing process is presented in Table 5 as a life cycle chart showing the verification activities for each phase. The success of conducting verification throughout the development cycle depends upon the existence of clearly defined and stated products at each development stage. The more formal and precise the statement of the development product, the more amenable it is to the analysis required to support verification. Many of the new system development methodologies encourage firm products even in the early development stages.

The recommended test process involves testing in every phase of the life cycle. During the requirements phase, the emphasis is upon validation to determine that the defined requirements meet the needs of the organization. During the design and program phases, the emphasis is on verification to ensure that the design and programs accomplish the defined requirements. During the test and installation phases, the emphasis is on inspection to determine that the implemented system meets the system specification. During the maintenance phases, the system will be retested to determine that the changes work and that the unchanged portion continues to work.

The following activities should be performed at each phase of the life cycle:

- Analyze the structures produced at this phase for internal testability and adequacy.

- Generate test sets based on the structure at this phase.

In addition, the following should be performed during design and programming:

- Determine that the structures are consistent with structures produced during previous phases.

- Refine or redefine test sets generated earlier.

Throughout the entire life cycle, neither development nor verification is a straight-line activity. Modifications or corrections to a structure at one phase will require modifications or reverification of structures produced during previous phases.

Requirements

The verification activities that accompany the problem definition and requirements analysis phase of software development are extremely significant. The adequacy of the requirements must be thoroughly analyzed and initial test cases generated with the expected (correct) responses. Developing scenarios of

expected system use helps to determine the test data and anticipated results. These tests form the core of the final test set. Generating these tests and the expected behavior of the system clarifies the requirements and helps guarantee that they are testable. Vague or requirements that are not testable leave the validity of the delivered product in doubt. Late discovery of requirements inadequacy can be very costly. A determination of the criticality of software quality attributes and the importance of validation should be made at this stage. Both product requirements and validation requirements should be established.

Design

Organization of the verification effort and test management activities should be closely integrated with preliminary design. The general testing strategy – including test methods and test evaluation criteria – is formulated, and a test plan is produced. If the project size or criticality warrants, an independent test team is organized. In addition, a test schedule with observable milestones is constructed. At this same time, the framework for quality assurance and test documentation should be established.
During detailed design, validation support tools should be acquired or developed and the test procedures themselves should be produced. Test data to exercise the functions introduced during the design process, as well as test cases based upon the structure of the system, should be generated. Thus, as the software development proceeds, a more effective set of test cases is built.

In addition to test organization and the generation of test cases, the design itself should be analyzed and examined for errors. Simulation can be used to verify properties of the system structures and subsystem interaction; the developers to verify the flow and logical structure of the system, while the test team should perform design inspections using design walkthroughs. Missing cases, faulty logic, module interface mismatches, data structure inconsistencies, erroneous I/O assumptions, and user interface inadequacies, are items of concern. The detailed design must prove to be internally coherent, complete, and consistent with the preliminary design and requirements.

Program (Build/Construction)

Actual testing occurs during the construction stage of development. Many testing tools and techniques exist for this stage of system development. Code walkthrough and code inspection are effective manual techniques. Static analysis techniques detect errors by analyzing program characteristics such as data flow and language construct usage. For programs of significant size, automated tools are required to perform this analysis. Dynamic analysis, performed as the code actually executes, is used to determine test coverage through various instrumentation techniques. Formal verification or proof techniques are used to provide further quality assurance.

Test Process

During the test process, careful control and management of test information is critical. Test sets, test results, and test reports should be catalogued and stored in a database. For all but very small systems, automated tools are required to do an adequate job – the bookkeeping chores alone become too large to handle manually. A test driver, test data generation aids, test coverage tools, test results management aids, and report generators are usually required.

Installation

The process of placing tested programs into production is an important phase normally executed within a narrow time span. Testing during this phase must ensure that the correct versions of the program are placed into production; that data if changed or added is correct; and that all involved parties know their new duties and can perform them correctly.

Maintenance

Over 50% of the life cycle costs of a software system are spent on maintenance. As the system is used, it is modified either to correct errors or to augment the original system. After each modification the system must be retested. Such retesting activity is termed regression testing. The goal of regression testing is to minimize the cost of system revalidation. Usually only those portions of the system impacted by the modifications are retested. However, changes at any level may necessitate retesting, re-verifying and updating documentation at all levels below it. For example, a design change requires design re-verification, unit retesting and subsystem retesting.

Test cases generated during system development are reused or used after appropriate modifications. The quality of the test documentation generated during system development and modified during maintenance will affect the cost of regression testing. If test data cases have been catalogued and preserved, duplication of effort will be minimized.

References

Guide – CSTE Common Body Of Knowledge, V6.1