

## Glossary of Software Testing/QA Terms

Source: The following definitions are taken from [GLOSSARY OF COMPUTERIZED SYSTEM AND SOFTWARE DEVELOPMENT TERMINOLOGY](http://www.fda.gov/ora/inspect_ref/igs/gloss.html) ([http://www.fda.gov/ora/inspect\\_ref/igs/gloss.html](http://www.fda.gov/ora/inspect_ref/igs/gloss.html))

**audit.** (1) (IEEE) An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria. See: functional configuration audit, physical configuration audit. (2) (ANSI) To conduct an independent review and examination of system records and activities in order to test the adequacy and effectiveness of data security and data integrity procedures, to ensure compliance with established policy and operational procedures, and to recommend any necessary changes. See: computer system audit, software audit.

**boundary value.** (1) (IEEE) A data value that corresponds to a minimum or maximum input, internal, or output value specified for a system or component. (2) A value which lies at, or just inside or just outside a specified range of valid input and output values.

**boundary value analysis.** (NBS) A selection technique in which test data are chosen to lie along "boundaries" of the input domain [or output range] classes, data structures, procedure parameters, etc. Choices often include maximum, minimum, and trivial values or parameters. This technique is often called stress testing. See: testing, boundary value.

**branch coverage.** (NBS) A test coverage criteria which requires that for each decision point each possible branch be executed at least once. Syn: decision coverage. Contrast with condition coverage, multiple condition coverage, path coverage, statement coverage. See: testing, branch.

**bug.** A fault in a program which causes the program to perform in an unintended or unanticipated manner. See: anomaly, defect, error, exception, fault.

**cause effect graph.** (Myers) A Boolean graph linking causes and effects. The graph is actually a digital-logic circuit (a combinatorial logic network) using a simpler notation than standard electronics notation.

**cause effect graphing.** (1) (NBS) Test data selection technique. The input and output domains are partitioned into classes and analysis is performed to determine which input classes cause which effect. A minimal set of inputs is chosen which will cover the entire effect set. (2) (Myers) A systematic method of generating test cases representing combinations of conditions. See: testing, functional.

**code inspection.** (Myers/NBS) A manual [formal] testing [error detection] technique where the programmer reads source code, statement by statement, to a group who ask questions analyzing the program logic, analyzing the code with respect to a checklist of historically common programming errors, and analyzing its compliance with coding standards. Contrast with code audit, code review, code walkthrough. This technique can also be applied to other software and configuration items. Syn: Fagan Inspection. See: static analysis.

**code review.** (IEEE) A meeting at which software code is presented to project personnel, managers, users, customers, or other interested parties for comment or approval. Contrast with code audit, code inspection, code walkthrough.

**code walkthrough.** (Myers/NBS) A manual testing [error detection] technique where program [source code] logic [structure] is traced manually [mentally] by a group with a small set of test cases, while the state of program variables is manually monitored, to analyze the programmer's logic and assumptions. Contrast with code audit, code inspection, code review. See: static analysis.

**coverage analysis.** (NIST) Determining and assessing measures associated with the invocation of program structural elements to determine the adequacy of a test run. Coverage analysis is useful when attempting to execute each statement, branch, path, or iterative structure in a program. Tools that capture this data and provide reports summarizing relevant information have this feature. See: testing, branch; testing, path; testing, statement.

**crash.** (IEEE) The sudden and complete failure of a computer system or component.

**criticality.** (IEEE) The degree of impact that a requirement, module, error, fault, failure, or other item has on the development or operation of a system. Syn: severity.

**cyclomatic complexity.** (1) (McCabe) The number of independent paths through a program. (2) (NBS) The cyclomatic complexity of a program is equivalent to the number of decision statements plus 1.

**error.** (ISO) A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. See: anomaly, bug, defect, exception, and fault

**error guessing.** (NBS) Test data selection technique. The selection criterion is to pick values that seem likely to cause errors. See: special test data; testing, special case.

**error seeding.** (IEEE) The process of intentionally adding known faults to those already in a computer program for the purpose of monitoring the rate of detection and removal, and estimating the number of faults remaining in the program. Contrast with mutation analysis.

**exception.** (IEEE) An event that causes suspension of normal program execution. Types include addressing exception, data exception, operation exception, overflow exception, protection exception, and underflow exception.

**failure.** (IEEE) The inability of a system or component to perform its required functions within specified performance requirements. See: bug, crash, exception, fault.

**fault.** An incorrect step, process, or data definition in a computer program which causes the program to perform in an unintended or unanticipated manner. See: bug, defect, error, exception.

**quality assurance.** (1) (ISO) The planned systematic activities necessary to ensure that a component, module, or system conforms to established technical requirements. (2) All actions that are taken to ensure that a development organization delivers products that meet performance requirements and adhere to standards and procedures. (3) The policy, procedures, and systematic actions established in an enterprise for the purpose of providing and maintaining some degree of confidence in data integrity and accuracy throughout the life cycle of the data, which includes input, update, manipulation, and output. (4) (QA) The actions, planned and performed, to provide confidence that all systems and components that influence the quality of the product are working as expected individually and collectively.

**quality assurance, software.** (IEEE) (1) A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements. (2) A set of activities designed to evaluate the process by which products are developed or manufactured.

**quality control.** The operational techniques and procedures used to achieve quality requirements.

**review.** (IEEE) A process or meeting during which a work product or set of work products, is presented to project personnel, managers, users, customers, or other interested parties for comment or approval. Types include code review, design review, formal qualification review, requirements review, test readiness review. Contrast with audit, inspection. See: static analysis.

**risk.** (IEEE) A measure of the probability and severity of undesired effects. Often taken as the simple product of probability and consequence.

**risk assessment.** (DOD) A comprehensive evaluation of the risk and its associated impact.

**software review.** (IEEE) An evaluation of software elements to ascertain discrepancies from planned results and to recommend improvement. This evaluation follows a formal process. Syn: software audit. See: code audit, code inspection, code review, code walkthrough, design review, specification analysis, static analysis

**static analysis.** (1) (NBS) Analysis of a program that is performed without executing the program. (2) (IEEE) The process of evaluating a system or component based on its form, structure, content, documentation. Contrast with dynamic analysis. See: code audit, code inspection, code review, code walk-through, design review, symbolic execution.

**test.** (IEEE) An activity in which a system or component is executed under specified conditions, the results are observed or recorded and an evaluation is made of some aspect of the system or component.

**testability.** (IEEE) (1) The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. (2) The degree to which a

requirement is stated in terms that permit establishment of test criteria and performance of tests to determine whether those criteria have been met.

**test case.** (IEEE) Documentation specifying inputs, predicted results, and a set of execution conditions for a test item. Syn: test case specification. See: test procedure.

**test case generator.** (IEEE) A software tool that accepts as input source code, test criteria, specifications, or data structure definitions; uses these inputs to generate test input data; and, sometimes, determines expected results. Syn: test data generator, test generator.

**test design.** (IEEE) Documentation specifying the details of the test approach for a software feature or combination of software features and identifying the associated tests. See: testing functional; cause effect graphing; boundary value analysis; equivalence class partitioning; error guessing; testing, structural; branch analysis; path analysis; statement coverage; condition coverage; decision coverage; multiple-condition coverage.

**test documentation.** (IEEE) Documentation describing plans for, or results of, the testing of a system or component. Types include test case specification, test incident report, test log, test plan, test procedure, test report.

**test driver.** (IEEE) A software module used to invoke a module under test and, often, provide test inputs, control and monitor execution, and report test results. Syn: **test harness.**

**test incident report.** (IEEE) A document reporting on any event that occurs during testing that requires further investigation.

**test item.** (IEEE) A software item which is the object of testing.

**test log.** (IEEE) A chronological record of all relevant details about the execution of a test.

**test phase.** (IEEE) The period of time in the software life cycle in which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied.

**test plan.** (IEEE) Documentation specifying the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks, responsibilities, required resources, and any risks requiring contingency planning. See: test design, validation protocol.

**test procedure.** (NIST) A formal document developed from a test plan that presents detailed instructions for the setup, operation, and evaluation of the results for each defined test. See: test case.

**test report.** (IEEE) A document describing the conduct and results of the testing carried out for a system or system component.

**test result analyzer.** A software tool used to test output data reduction, formatting, and printing.

**testing.** (IEEE) (1) The process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component. (2) The process of analyzing a software item to detect the differences between existing and required conditions, i.e. bugs, and to evaluate the features of the software items. See: dynamic analysis, static analysis

**testing, acceptance.** (IEEE) Testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. Contrast with testing, development; testing, operational.

**testing, alpha [].** (Pressman) Acceptance testing performed by the customer in a controlled environment at the developer's site. The software is used by the customer in a setting approximating the target environment with the developer observing and recording errors and usage problems.

**testing, assertion.** (NBS) A dynamic analysis technique which inserts assertions about the relationship between program variables into the program code. The truth of the assertions is determined as the program executes.

**testing, beta [1].** (1) (Pressman) Acceptance testing performed by the customer in a live application of the software, at one or more end user sites, in an environment not controlled by the developer. (2) For medical device software such use may require an Investigational Device Exemption [IDE] or Institutional Review Board [IRB] approval.

**testing, boundary value.** A testing technique using input values at, just below, and just above, the defined limits of an input domain; and with input values causing outputs to be at, just below, and just above, the defined limits of an output domain. See: boundary value analysis; testing, stress.

**testing, branch.** (NBS) Testing technique to satisfy coverage criteria which require that for each decision point, each possible branch [outcome] be executed at least once. Contrast with testing, path; testing, statement. See: branch coverage.

**testing, compatibility.** The process of determining the ability of two or more systems to exchange information. In a situation where the developed software replaces an already working program, an investigation should be conducted to assess possible comparability problems between the new software and other programs or systems.

**testing, exhaustive.** (NBS) Executing the program with all possible combinations of values for program variables. Feasible only for small, simple programs.

**testing, functional.** (IEEE) (1) Testing that ignores the internal mechanism or structure of a system or component and focuses on the outputs generated in response to selected inputs and execution conditions. (2) Testing conducted to evaluate the compliance of a system or component with specified functional requirements and corresponding predicted results. Syn: black-box testing, input/output driven testing. Contrast with testing, structural.

**testing, integration.** (IEEE) An orderly progression of testing in which software elements, hardware elements, or both are combined and tested, to evaluate their interactions, until the entire system has been integrated.

**testing, interface.** (IEEE) Testing conducted to evaluate whether systems or components pass data and control correctly to one another. Contrast with testing, unit; testing, system. See: testing, integration.

**testing, mutation.** (IEEE) A testing methodology in which two or more program mutations are executed using the same test cases to evaluate the ability of the test cases to detect differences in the mutations.

**testing, operational.** (IEEE) Testing conducted to evaluate a system or component in its operational environment. Contrast with testing, development; testing, acceptance; See: testing, system.

**testing, parallel.** (ISO) Testing a new or an altered data processing system with the same source data that is used in another system. The other system is considered as the standard of comparison. Syn: parallel run.

**testing, path.** (NBS) Testing to satisfy coverage criteria that each logical path through the program be tested. Often paths through the program are grouped into a finite set of classes. One path from each class is then tested. Syn: path coverage. Contrast with testing, branch; testing, statement; branch coverage; condition coverage; decision coverage; multiple condition coverage; statement coverage.

**testing, performance.** (IEEE) Functional testing conducted to evaluate the compliance of a system or component with specified performance requirements.

**testing, qualification.** (IEEE) Formal testing, usually conducted by the developer for the consumer, to demonstrate that the software meets its specified requirements. See: testing, acceptance; testing, system.

**testing, regression.** (NIST) Rerunning test cases which a program has previously executed correctly in order to detect errors spawned by changes or corrections made during software development and maintenance.

**testing, statement.** (NIST) Testing to satisfy the criterion that each statement in a program be executed at least once during program testing. Syn: statement coverage. Contrast with testing, branch; testing, path; branch coverage; condition coverage; decision coverage; multiple condition coverage; path coverage.

**testing, storage.** This is a determination of whether or not certain processing conditions use more storage [memory] than estimated.

**testing, stress.** (IEEE) Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements. Syn: testing, boundary value.

**testing, structural.** (1) (IEEE) Testing that takes into account the internal mechanism [structure] of a system or component. Types include branch testing, path testing, statement testing. (2) Testing to insure each program statement is made to execute during testing and that each program statement performs its intended function. Contrast with functional testing. Syn: white-box testing, glass-box testing, logic driven testing.

**testing, system.** (IEEE) The process of testing an integrated hardware and software system to verify that the system meets its specified requirements. Such testing may be conducted in both the development environment and the target environment.

**testing, unit.** (1) (NIST) Testing of a module for typographic, syntactic, and logical errors, for correct implementation of its design, and for satisfaction of its requirements. (2) (IEEE) Testing conducted to verify the implementation of the design for one software element; e.g., a unit or module; or a collection of software elements. Syn: component testing.

**testing, usability.** Tests designed to evaluate the machine/user interface. Are the communication device(s) designed in a manner such that the information is displayed in a understandable fashion enabling the operator to correctly interact with the system?

**testing, volume.** Testing designed to challenge a system's ability to manage the maximum amount of data over a period of time. This type of testing also evaluates a system's ability to handle overload situations in an orderly fashion.

**traceability matrix.** (IEEE) A matrix that records the relationship between two or more products; e.g., a matrix that records the relationship between the requirements and the design of a given software component. See: traceability, traceability analysis.

**usability.** (IEEE) The ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component.

**validation.** (1) (FDA) Establishing documented evidence which provides a high degree of assurance that a specific process will consistently produce a product meeting its predetermined specifications and quality attributes. Contrast with data validation.

**validation, verification, and testing.** (NIST) Used as an entity to define a procedure of review, analysis, and testing throughout the software life cycle to discover errors, determine functionality, and ensure the production of quality software.

**verification, software.** (NBS) In general the demonstration of consistency, completeness, and correctness of the software at each stage and between each stage of the development life cycle. See: validation, software.