ISTQB GLOSSARY

Version 4.2 2024-01-20

Based on ”ISTQB Standard Glossary of Terms Used in Software Testing”

Scope

This glossary is a copy of the ISTQB Standard Glossary of Terms Used in Software Testing issued by the ISTQB Glossary Working Group.

The ISTQB Standard Glossary of Terms Used in Software Testing contains the definitions of testing terms used in the different ISTQB syllabi. This includes all terms stated as keywords in the ISTQB syllabi, as well as other terms of major importance.

The ISTQB Glossary focuses on terms that have a specific meaning in testing. Some related non-testing terms are also included if they play a major role in testing, such as terms used in software quality assurance and software lifecycle models. However, most terms of other software engineering disciplines are not covered in this document, even if they are used in various ISTQB syllabi.

Purpose of the ISTQB Glossary

The ISTQB Glossary has two main objectives:

- Support the ISTQB syllabi by defining the terms used in the various syllabi consistently;
- Support communication within the international testing community and with its stakeholders by providing a standard testing vocabulary.

In compiling this Glossary, the ISTQB Glossary Working Group has sought the views and comments of a broad spectrum of opinion in industry, commerce and government bodies and organizations, with the aim of producing an international testing standard that would gain wide acceptance. Total agreement will rarely, if ever, be achieved in compiling a document of this nature. Contributions to this glossary have been received from testing communities from all over the world.

Being written in English, the current version of the Glossary is designed to also support other languages. ISTQB Member Boards are encouraged to incorporate their translations.

Glossary Structure

The glossary has been arranged in a single section of terms and their definitions, ordered alphabetically. For each term, the following additional attributes are shown where applicable:

- Ref: without the addition of “after”, e.g., ISO 25010, this means that the exact definition of the reference is used. In case of minor changes used to adapt the definition to the context of the ISTQB Glossary, the addition “after” is used, e.g., Ref: After ISO 25010. The complete list of references used in the ISTQB Glossary is listed below.
- Synonym: Some terms are preferred to other synonymous ones, in which case, the preferred term appears as an entry, with the synonyms indicated.
- See also: These entries contain cross-references to related terms. Such cross-references are indicated for relationships such as broader term to a narrower term and overlapping meaning between two terms.

Acknowledgements

This Glossary has been produced by the Glossary Working Group of the International Software Testing Qualifications Board (ISTQB).

At the time the Glossary version 3.2 was completed the Glossary Working Group had the following members (alphabetic order):

Tobias Ahlgren (Sweden), Vineta Arnicane (Latvia), Armin Beer (Austria), Armin Born (Switzerland), Mette Bruhn-Pedersen (Denmark), Gergory Collina (USA), Matthias Daigl (Germany), Ernst Dűring (Norway), George Fialkovitz (Brazil), Matthias Hamburg (Chair, Germany), Tamás Horváth (Hungary), Leanne Howard (Australia), Ian Howles (Great Britain), Marek Majernik (Slovakia), Gustavo Marquez Sosa (Spain), Judy McKay (USA), Gary Mogyorodi (Vice-Chair, Canada), Ana Paiva (Portugal), Juha Pomppu (Finnland), Meile Posthuma (Netherlands). Adam Roman (Poland), Lucjan Stapp (Poland), Karolina Zmitrowitz (Poland).

It is our concern to recognize the pioneering merits of Erik van Veenendaal who has designed the first version of this Glossary and who conducted the Glossary Working Group during many years, from its beginnings until 2014.

Our special thanks go to Nicholas Humphries for the development of the interactive application.

Many more people, who are not mentioned here by name, have contributed to different versions of this Glossary. The editors would like to thank them all for their contributions.

Definitions

A/B testing: A statistical testing approach to determine which of two systems or components performs better.
Reference: After ISO 29119-11

abnormal end: The unintended termination of the execution of a component or system prior to completion.
Reference: After ISO 24765
Synonym: abnormal termination

abuse case: A use case in which some actors with malicious intent are causing harm to the system or to other actors.
See also: use case

acceptance criteria: The criteria that a component or system must satisfy in order to be accepted by a user, customer, or other authorized entity.
Reference: ISO 24765

acceptance test-driven development: (ATDD) A collaboration-based test-first approach that defines acceptance tests in the stakeholders' domain language.

acceptance testing: A test level that focuses on determining whether to accept the system.

accessibility: The degree to which a component or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.
Reference: After ISO 25010

account harvesting: The process of obtaining user account information based on trial and error with the intention of using that information in a security attack.

accountability: The degree to which the actions of an entity can be traced uniquely to that entity.
Reference: After ISO 25010

actual result: The behavior produced/observed when a component or system is tested.

ad hoc review: A review technique performed informally without a structured process.
Reference: After ISO 20246

ad hoc testing: Informal testing performed without test analysis and test design.

adaptability: The degree to which a component or system can be adapted for different or evolving hardware, software or other operational or usage environments.
Reference: After ISO 25010

adversarial example: An input to an ML model created by applying small perturbations to a working example that results in the model outputting an incorrect result with high confidence.
Reference: ISO 29119-11

adversarial testing: A test technique based on the attempted creation and execution of adversarial examples to identify defects in an ML model.
Reference: After ISO 29119-11

Agile Manifesto: A statement on the values that underpin Agile software development. The values are: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, responding to change over following a plan.

Agile software development: A group of software development methodologies based on iterative incremental development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.

Agile test leader: A leadership role that serves agile teams, championing testing and quality at the organizational level.
Synonym: quality coach, agile test manager, quality leader

Agile test team leader: A role that is responsible for maintaining solution quality within an agile delivery team.

alpha testing: A type of acceptance testing performed in the developer's test environment by roles outside the development organization.

analytical test strategy: A test strategy whereby the test team analyzes the test basis to identify the test conditions to cover.

analyzability: The degree to which an assessment can be made for a component or system of either the impact of one or more intended changes, the diagnosis of deficiencies or causes of failures, or the identification of parts to be modified.
Reference: After ISO 25010

anomaly: A condition that deviates from expectation.
Reference: ISO 24765

anti-malware: Software that is used to detect and inhibit malware.
See also: malware

API testing: Testing performed by submitting commands to the software under test using programming interfaces of the application directly.

application programming Interface: (API) A type of interface in which the components or systems involved exchange information in a defined formal structure.

appropriateness recognizability: The degree to which users can recognize whether a component or system is appropriate for their needs.
Reference: After ISO 25010

atomic condition: A condition that does not contain logical operators.

attack vector: A path or means by which an attacker can gain access to a system for malicious purposes.

attacker: A person or process that attempts to access data, functions or other restricted areas of the system without authorization, potentially with malicious intent.
See also: hacker

audio testing: Testing to determine if the game music and sound effects will engage the user in the game and enhance the game play.

audit: An independent examination of a work product or process performed by a third party to assess whether it complies with specifications, standards, contractual agreements, or other criteria.
Reference: After ISO 24765

authentication: A procedure determining whether a person or a process is, in fact, who or what it is declared to be.
See also: authorization

authenticity: The degree to which the identity of a subject or resource can be proved to be the one claimed.
Reference: ISO 25010

authorization: Permission given to a user or process to access resources.
See also: authentication

automation code defect density: Defect density of a component of the test automation code.
See also: defect density

automotive safety integrity level: (ASIL) One of four levels that specify the item's or element's necessary requirements of ISO 26262 and safety measures to avoid an unreasonable residual risk.
Reference: ISO 26262

automotive SPICE: (ASPICE) A process reference model and an associated process assessment model in the automotive industry that conforms with the requirements of ISO/IEC 33002:2015.
Reference: Automotive SPICE

availability: The degree to which a component or system is operational and accessible when required for use.
Reference: After ISO 25010

back-to-back testing: Testing to compare two or more variants of a test item or a simulation model of the same test item by executing the same test cases on all variants and comparing the results.
Reference: Spillner

behavior-driven development: (BDD) A collaborative approach to development in which the team is focusing on delivering expected behavior of a component or system for the customer, which forms the basis for testing.

beta testing: A type of acceptance testing performed at an external site to the developer's test environment by roles outside the development organization.

black-box test technique: A test technique based on an analysis of the specification of a component or system.

black-box testing: Testing based on an analysis of the specification of the component or system.

botnet: A network of compromised computers, called bots or robots, which is controlled by a third party and used to transmit malware or spam, or to launch attacks.

boundary value: A minimum or maximum value of an ordered equivalence partition.

boundary value analysis: A black-box test technique in which test cases are designed based on boundary values.

branch: A transfer of control between two nodes in the control flow graph of a test item.

branch coverage: The coverage of branches in a control flow graph.

branch testing: A white-box test technique in which the test conditions are branches.

bug hunting: An approach to testing in which gamification and awards for defects found are used as a motivator.

build verification test: (BVT) A set of automated tests which validates the integrity of each new build and verifies its key/core functionality, stability and testability.

built-in quality: A set of practices to ensure that each solution meets quality standards throughout each increment of development, focusing on constructive quality assurance as a shared responsibility.
Reference: After SAFe

Capability Maturity Model Integration: (CMMI) A framework that describes the key elements of an effective product development and maintenance process. The Capability Maturity Model Integration covers best-practices for planning, engineering and managing product development and maintenance.
Reference: CMMI

capacity: The degree to which the maximum limits of a component or system parameter meet requirements.
Reference: After ISO 25010

capacity testing: Testing to evaluate the capacity of a system.

capture/playback: A test automation approach in which inputs to the test object are recorded during manual testing to generate automated test scripts that can be executed later.

causal loop diagram: A graphical representation used to visualize cause-effect relationships and feedback loops in a system.
Reference: After Sterman00

cause-effect diagram: A graphical representation used to organize and display the interrelationships of various possible root causes of a problem. Possible causes of a real or potential defect or failure are organized in categories and subcategories in a horizontal tree-structure, with the (potential) defect or failure as the root node.
Reference: After Juran

cause-effect graph: A graphical representation of logical relationships between inputs (causes) and their associated outputs (effects) of a test object.

certification: The process of confirming that a component, system or person complies with specified requirements.

change-related testing: A type of testing initiated by modification to a component or system.

checklist-based review: A review technique guided by a list of questions or required attributes.
Reference: ISO 20246

checklist-based testing: An experience-based test technique in which test cases are designed to exercise the items of a checklist.

classification tree: A tree diagram representing test data domains of a test object.

classification tree technique: A black-box test technique in which test cases are designed using a classification tree.
Reference: Grochtmann

CLI testing: Testing performed by submitting commands to the software under test using a dedicated command-line interface.

closed-loop-system: A system in which the controlling action or input is dependent on the output or changes in output.
Reference: Bakshi

code injection: A type of security attack performed by inserting malicious code at an interface into an application to exploit poor handling of untrusted data.
See also: malware scanning, SQL injection

coding standard: A standard that describes the characteristics of a design or a design description of data or program components.
Reference: ISO 24765

coexistence: The degree to which a component or system can perform its required functions while sharing an environment and resources with other components or systems without a negative impact on any of them.
Reference: After ISO 25010
Synonym: co-existence

collaboration-based test approach: An approach to testing that focuses on defect avoidance by collaborating among stakeholders.

combinatorial testing: A black-box test technique in which test conditions are specific combinations of values of several parameters.
See also: pairwise testing, classification tree technique

command-line interface: (CLI) A type of interface in which the information is passed in form of command lines.

commercial off-the-shelf: (COTS) A type of product developed in an identical format for a large number of customers in the general market.

compatibility: The degree to which a component or system can exchange information with other components or systems, and/or perform its required functions while sharing the same hardware or software environment.
Reference: After ISO 25010

complexity: The degree to which the design or code of a component or system is difficult to understand.
See also: cyclomatic complexity

compliance: Adherence of a work product to standards, conventions or regulations in laws and similar prescriptions.
Reference: IREB Glossary

compliance testing: Testing to determine the compliance of the component or system.

component: A part of a system that can be tested in isolation.

component integration testing: The integration testing of components
Synonym: module integration testing, unit integration testing

component testing: A test level that focuses on individual hardware or software components.

computer forensics: The practice of determining how a security attack has succeeded and assessing the damage caused.

concurrency: The simultaneous execution of multiple independent threads by a component or system.

concurrency testing: Testing to evaluate if a component or system involving concurrency behaves as specified.

condition coverage: The coverage of condition outcomes.

condition testing: A white-box test technique in which test conditions are outcomes of atomic conditions.

confidence interval: In managing project risks, the period of time within which a contingency action must be implemented in order to be effective in reducing the impact of the risk.

confidentiality: The degree to which a component or system ensures that data are accessible only to those authorized to have access.
Reference: After ISO 25010

configuration management: A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify that it complies with specified requirements.

confirmation testing: A type of change-related testing performed after fixing a defect to confirm that a failure caused by that defect does not reoccur.

connectivity: The degree to which a component or system can connect to other components or systems.
Reference: After ISO 2382

consultative test strategy: A test strategy whereby the test team relies on the input of one or more key stakeholders to determine the details of the strategy.

context of use: Users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a software product is used.
Reference: ISO 9241-11

continuous integration: An automated software development procedure that merges, integrates and tests all changes as soon as they are committed.

continuous testing: An approach that involves a process of testing early, testing often, test everywhere, and automate to obtain feedback on the business risks associated with a software release candidate as rapidly as possible.

contractual acceptance testing: A type of acceptance testing performed to verify whether a system satisfies its contractual requirements.

control chart: A statistical process control tool used to monitor a process and determine whether it is statistically controlled. It graphically depicts the average value and the upper and lower control limits (the highest and lowest values) of a process.

control flow: The sequence in which operations are performed by a business process, component or system.
Reference: After ISO 29119-4

control flow analysis: A type of static analysis based on a representation of unique paths for executing a component or system.

control flow testing: A white-box test technique in which test cases are designed based on control flows.

convergence metric: A metric that shows progress toward a defined criterion, e.g., convergence of the total number of tests executed to the total number of tests planned for execution.

cost of quality: The total costs incurred on quality activities and issues and often split into prevention costs, appraisal costs, internal failure costs and external failure costs.

coverage: The degree to which specified coverage items are exercised by a test suite, expressed as a percentage.
Reference: After ISO 29119-1

coverage criteria: The criteria to define the test coverage items required to reach a test objective.
See also: coverage item

coverage item: An attribute or combination of attributes derived from one or more test conditions by using a test technique.
Reference: After ISO 29119-1

Critical Testing Processes: (CTP) A content-based model for test process improvement built around twelve critical processes. These include highly visible processes, by which peers and management judge competence and mission-critical processes in which performance affects the company's profits and reputation.
See also: content-based model

cross-browser compatibility: The degree to which a website or web application can function across different browsers and degrade gracefully when browser features are absent or lacking.

cross-site scripting: (XSS) A vulnerability that allows attackers to inject malicious code into an otherwise benign website.
Reference: NIST.IR.7298

crowd testing: A test approach to testing in which testing is distributed to a large group of testers.

custom tool: A software tool developed specifically for a set of users or customers.

cyclomatic complexity: The maximum number of linear, independent paths through a program.
Reference: After McCabe

dashboard: A representation of dynamic measurements of operational performance for some organization or activity, using metrics represented via metaphors such as visual dials, counters, and other devices resembling those on the dashboard of an automobile, so that the effects of events or activities can be easily understood and related to operational goals.
See also: corporate dashboard, scorecard

data flow analysis: A type of static analysis based on the lifecycle of variables.

data obfuscation: Data transformation that makes it difficult for a human to recognize the original data.

data privacy: The protection of personally identifiable information or otherwise sensitive information from undesired disclosure.

data-driven testing: A scripting technique that uses data files to contain the test data and expected results needed to execute the test scripts.

debugging: The process of finding, analyzing and removing the causes of failures in a component or system.

decision coverage: The coverage of decision outcomes.

decision table testing: A black-box test technique in which test cases are designed to exercise the combinations of conditions and the resulting actions shown in a decision table.

decision testing: A white-box test technique in which test cases are designed to execute decision outcomes.

defect: An imperfection or deficiency in a work product where it does not meet its requirements or specifications.
Reference: After ISO 24765

defect density: The number of defects per unit size of a work product.
Reference: After ISO 24765

Defect Detection Percentage: (DDP) The number of defects found by a test level, divided by the number found by that test level and any other means afterwards.
See also: escaped defect

defect management: The process of recognizing, recording, classifying, investigating, resolving and disposing of defects.

defect management committee: A cross-functional team of stakeholders who manage reported defects from initial detection to ultimate resolution (defect removal, defect deferral, or report cancellation). In some cases, the same team as the configuration control board.

defect report: Documentation of the occurrence, nature, and status of a defect.

defect taxonomy: A list of categories designed to identify and classify defects.
Synonym: bug taxonomy

defect-based test technique: A test technique in which test cases are developed from what is known about a specific defect type.
Synonym: defect-based technique

definition-use pair: The association of a definition of a variable with the subsequent use of that variable.

demilitarized zone: (DMZ) A physical or logical subnetwork that contains and exposes an organization's external-facing services to an untrusted network, commonly the Internet.
See also: network zone

denial of service: (DoS) A security attack that is intended to overload the system with requests such that legitimate requests cannot be serviced.

device-based testing: A type of testing in which test suites are executed on physical or virtual devices.

driver: A component or tool that temporarily replaces another component and controls or calls a test item in isolation.

dynamic analysis: The process of evaluating a component or system based on its behavior during execution.
Reference: After ISO 24765

dynamic testing: Testing that involves the execution of the test item.
Reference: After ISO 29119-1

effectiveness: The extent to which correct and complete goals are achieved.
Reference: ISO 9241
See also: efficiency

efficiency: The degree to which resources are expended in relation to results achieved.
Reference: IREB Glossary

EFQM model: A management framework that supports organisations in managing change and improving performance.

emulator: Software used during testing that mimics the behavior of hardware.
Reference: ISO 24765

encryption: The process of encoding information so that only authorized parties can retrieve the original information, usually by means of a specific decryption key or process.

end-to-end testing: A test type in which business processes are tested from start to finish under production-like circumstances.

endurance testing: Testing to determine the stability of a system under a significant load over a significant period of time within the system's operational context.

entry criteria: The set of conditions for officially starting a defined task.
Reference: Gilb and Graham
See also: exit criteria

environment model: An abstraction of the real environment of a component or system including other components, processes, and environment conditions, in a real-time simulation.
Reference: Wallentowitz

epic: A large user story that cannot be delivered as defined within a single iteration or is large enough that it can be split into smaller user stories.
Reference: Agile Alliance

equivalence partition: A subset of the value domain of a variable within a component or system in which all values are expected to be treated the same based on the specification.
Reference: Efter ISO 29119

equivalence partitioning: A black-box test technique in which test conditions are equivalence partitions exercised by one representative member of each partition.
Reference: After ISO 29119-1

equivalent manual test effort: (EMTE) Effort required for running tests manually.

ergonomics testing: Testing to determine whether a component or system and its input devices are being used properly with correct posture.

error: A human action that produces an incorrect result.
Reference: ISO 24765

error guessing: A test technique in which tests are derived on the basis of the tester's knowledge of past failures, or general knowledge of failure modes.
Reference: ISO 29119-1

escaped defect: A defect that was not detected by a testing activity that is supposed to find that defect.

ethical hacker: A security tester using hacker techniques.

exhaustive testing: A test approach in which the test suite comprises all combinations of input values and preconditions.

exit criteria: The set of conditions for officially completing a defined task.
Reference: After Gilb and Graham

expected result: The observable predicted behavior of a test item under specified conditions based on its test basis.
Reference: After ISO 29119-1

experience-based test technique: A test technique based on the tester's experience, knowledge and intuition.

experience-based testing: Testing based on the tester's experience, knowledge and intuition.

expert usability review: An informal usability review in which the reviewers are experts. Experts can be usability experts or subject matter experts, or both.
See also: informal review

exploratory testing: An approach to testing in which the testers dynamically design and execute tests based on their knowledge, exploration of the test item and the results of previous tests.
Reference: After ISO 29119-1

failed: The status of a test result if the actual result does not match the expected result.

failover: The backup operational mode in which the functions of a system that becomes unavailable are assumed by a secondary system.

failure: An event in which a component or system does not perform a required function within specified limits.
Reference: After ISO 24765

failure mode: The physical or functional manifestation of a failure.
Reference: ISO 24765

Failure Mode and Effect Analysis: (FMEA) A systematic approach to risk identification and analysis of identifying possible modes of failure and attempting to prevent their occurrence.
See also: Failure Mode, Effect and Criticality Analysis

failure rate: The ratio of the number of failures of a given category to a given unit of measure.
Reference: ISO 24765

false-negative result: A test result which fails to identify the presence of a defect that is actually present in the test object.

false-positive result: A test result in which a defect is reported although no such defect actually exists in the test object.

fault injection: The process of intentionally adding a defect to a component or system to determine whether it can detect and possibly recover from it.

fault seeding: The process of intentionally adding defects to a component or system to monitor the rate of detection and removal, and to estimate the number of defects remaining.
Reference: After ISO 24765

fault tolerance: The degree to which a component or system operates as intended despite the presence of hardware or software faults.
Reference: After ISO 25010
Synonym: SFTA (Software Fault Tree Analysis)

fault tree analysis: (FTA) A technique for analyzing the causes of failures that uses a hierarchical model of events and their logical relationships.

feature-driven development: An iterative and incremental software development process driven from a client-valued functionality (feature) perspective. Feature-driven development is mostly used in Agile software development.
See also: Agile software development

field testing: A type of testing conducted to evaluate the system behavior under productive connectivity conditions in the field.

finding: A result of an evaluation that identifies some important issue, problem, or opportunity.

firewall: A component or set of components that controls incoming and outgoing network traffic based on predetermined security rules.

follow-up test case: A test case generated by applying a metamorphic relation to a source test case during metamorphic testing.

formal review: A type of review that follows a defined process with a formally documented output.
Reference: ISO 20246

formative evaluation: A type of evaluation designed and used to improve the quality of a component or system, especially when it is still being designed.
See also: summative evaluation

functional appropriateness: The degree to which the functions facilitate the accomplishment of specified tasks and objectives.
Reference: ISO 25010

functional completeness: The degree to which the set of functions covers all the specified tasks and user objectives.
Reference: ISO 25010

functional correctness: The degree to which a component or system provides the correct results with the needed degree of precision.
Reference: After ISO 25010

functional safety: The absence of unreasonable risk due to hazards caused by malfunctioning behavior of Electric/Electronic(E/E) - Systems.
Reference: ISO 26262

functional suitability: The degree to which a component or system provides functions that meet stated and implied needs when used under specified conditions.
Reference: After ISO 25010

functional testing: Testing performed to evaluate if a component or system satisfies functional requirements.
Reference: After ISO 24765

fuzz testing: A software testing technique used to discover security vulnerabilities by inputting massive amounts of random data, called fuzz, to the component or system.

generic test automation architecture: Representation of the layers, components, and interfaces of a test automation architecture, allowing for a structured and modular approach to implement test automation.

graphical user interface: (GUI) A type of interface that allows users to interact with a component or system through graphical icons and visual indicators.

GUI testing: Testing performed by interacting with the software under test via the graphical user interface.

hacker: A person or organization who is actively involved in security attacks, usually with malicious intent.
See also: attacker

hardware in the loop: (HiL) Dynamic testing performed using real hardware with integrated software in a simulated environment.
Reference: Automotive SPICE

hashing: Transformation of a variable length string of characters into a usually shorter fixed-length value or key. Hashed values, or hashes, are commonly used in table or database lookups. Cryptographic hash functions are used to secure data.

heuristic: A generally recognized rule of thumb that helps to achieve a goal.

heuristic evaluation: A usability review technique that evaluates a work product by using a set of heuristics.

high-level test case: A test case with abstract preconditions, input data, expected results, postconditions, and actions (where applicable).

human-centered design: An approach to design that aims to make software products more usable by focusing on the use of the software products and applying human factors, ergonomics, and usability knowledge and techniques.
Reference: ISO 9241-210

hyperlink: A pointer within a web page that leads to other web pages.

impact analysis: The identification of all work products affected by a change, including an estimate of the resources needed to accomplish the change.
Reference: After ISO 24765

incremental development model: A type of software development lifecycle model in which the component or system is developed through a series of increments.
Reference: After PMBOK

independence of testing: Separation of responsibilities, which encourages the accomplishment of objective testing.
Reference: After DO-178C

independent test lab: (ITL) An organization responsible to test and certify that the software, hardware, firmware, platform, and operating system follow all the jurisdictional rules for each location where the product will be used.

informal review: A type of review that does not follow a defined process and has no formally documented output.

information assurance: Measures that protect and defend information and information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation. These measures include providing for restoration of information systems by incorporating protection, detection, and reaction capabilities.
Reference: NIST.IR.7298

input data testing: A test level that focuses on the quality of the data used for training and prediction by ML models.

insider threat: A security threat originating from within the organization, often by an authorized system user.

insourced testing: Testing performed by people who are co-located with the project team but are not fellow employees.

inspection: A type of formal review that uses defined team roles and measurement to identify defects in a work product, and improve the review process and the software development process.
Reference: After ISO 20246

installability: The degree to which a component or system can be successfully installed and/or uninstalled in a specified environment.
Reference: After ISO 25010

integration testing: A test level that focuses on interactions between components or systems.

integrity: The degree to which a component or system allows only authorized access and modification to a component, a system or data.
Reference: After ISO 25010

interface testing: A type of integration testing performed to determine whether components or systems pass data and control correctly to one another.
Reference: After ISO 24765

internationalization: The process of making a system suitable for international user groups.

interoperability: The degree to which two or more components or systems can exchange information and use the information that has been exchanged.
Reference: After ISO 25010

intrusion detection system: (IDS) A system which monitors activities on the 7 layers of the OSI model from network to application level, to detect violations of the security policy.
See also: malware scanning

iterative development model: A type of software development lifecycle model in which the component or system is developed through a series of repeated cycles.

keyword-driven testing: A scripting technique in which test scripts contain high-level keywords and supporting files that contain low-level scripts that implement those keywords.

learnability: The degree to which a component or system can be used by specified users to achieve specified goals of learning with satisfaction and freedom from risk in a specified context of use.
Reference: After ISO 25010

level of intrusion: The level to which a test object is modified by adjusting it for testability.

level test plan: A test plan that typically addresses one test level.
See also: test plan

linear scripting: A simple scripting technique without any control structure in the test scripts.

load generation: The process of simulating a defined set of activities at a specified load to be submitted to a component or system.
See also: load testing

load generator: A tool that generates a load for a system under test.

load management: The control and execution of load generation, performance monitoring, and reporting of the component or system.

load profile: Documentation defining a designated number of virtual users who process a defined set of transactions in a specified time period that a component or system being tested may experience in production.

load testing: A type of performance testing conducted to evaluate the behavior of a component or system under varying loads, usually between anticipated conditions of low, typical, and peak usage.
Reference: After ISO 29119-1
See also: performance testing, stress testing

localization: The process of making a system suitable for a specific user group.

low-level test case: A test case with concrete values for preconditions, input data, expected results, postconditions, and a detailed description of actions (where applicable).
See also: high-level test case

maintainability: The degree to which a component or system can be modified by the intended maintainers.
Reference: After ISO 25010

maintenance: The process of modifying a component or system after delivery to correct defects, improve quality characteristics, or adapt to a changed environment.
Reference: After ISO 24765

maintenance testing: Testing the changes to an operational system or the impact of a changed environment to an operational system.

malware: Software that is intended to harm a system or its components.

malware scanning: Static analysis aiming to detect and remove malicious code received at an interface.
See also: intrusion detection system

management review: A systematic evaluation of software acquisition, supply, development, operation, or maintenance process, performed by or on behalf of management that monitors progress, determines the status of plans and schedules, confirms requirements and their system allocation, or evaluates the effectiveness of management approaches to achieve fitness for purpose.
Reference: After ISO 24765, IEEE 1028

manufacturing-based quality: A view of quality measured by the degree that a product or service conforms to its intended design and requirements based on the process used.
Reference: After Garvin
See also: product-based quality, transcendent-based quality, user-based quality, value-based quality

master test plan: A test plan that is used to coordinate multiple test levels or test types.
See also: test plan

math testing: Testing to determine the correctness of the pay table implementation, the random number generator results, and the return to player computations.

maturity: (1) The capability of an organization with respect to the effectiveness and efficiency of its processes and work practices. (2) The degree to which a component or system meets needs for reliability under normal operation.
Reference: ISO 25010

MBT model: Any model used in model-based testing.

mean time between failures: (MTBF) The average time between failures of a component or system.

mean time to failure: (MTTF) The average time from the start of operation to a failure for a component or system.

mean time to repair: (MTTR) The average time a component or system will take to recover from a failure.

measurement: The process of assigning a number or category to an entity to describe an attribute of that entity.
Reference: After ISO 24765

memory leak: A memory access failure due to a defect in a program's dynamic store allocation logic that causes it to fail to release memory after it has finished using it.

metamorphic relation: (MR) A description of how a change in the test inputs from the source test case to the follow-up test case affects a change in the expected outputs from the source test case to the follow-up test case.
Reference: ISO 29119-11

metamorphic testing: (MT) A test technique in which the inputs and expected results are extrapolated from a passing test case using a metamorphic relation.

method table: A table containing different test approaches, testing techniques and test types that are required depending on the Automotive Safety Integrity Level (ASIL) and on the context of the test object.
Reference: ISO 26262

methodical test strategy: A test strategy whereby the test team uses a pre-determined set of test conditions such as a quality standard, a checklist, or a collection of generalized, logical test conditions which may relate to a particular domain, application or type of testing.

metric: A measurement scale and the method used for measurement.

ML functional performance: The degree to which an ML model meets ML functional performance criteria.

ML functional performance criteria: Criteria based on ML functional performance metrics used as a basis for model evaluation, tuning and testing.

ML functional performance metrics: A set of measures that relate to the functional correctness of an ML system.

ML model: An implementation of machine learning (ML) that generates a prediction, classification or recommendation based on input data.

ML model testing: A test level that focuses on the ability of an ML model to meet required ML functional performance criteria and non-functional criteria.
Reference: ISO 29119-11

model coverage: The coverage of model elements.

model in the loop: (MiL) Dynamic testing performed using a simulation model of the system in a simulated environment.
Reference: Automotive SPICE

model-based test strategy: A test strategy whereby the test team derives testware from models.

model-based testing: (MBT) Testing based on or involving models.

moderator: (1) The person responsible for running review meetings. (2) The person who performs a usability test session.

modifiability: The degree to which a component or system can be modified without degrading its quality.
Reference: After ISO 25010

modified condition/decision coverage: (MC/DC) The coverage of all outcomes of the atomic conditions that independently affect the overall decision outcome.

modified condition/decision testing: A white-box test technique in which test cases are designed to exercise outcomes of atomic conditions that independently affect a decision outcome.

modularity: The degree to which a system is composed of discrete components such that a change to one component has minimal impact on other components.
Reference: After ISO 25010

monitoring tool: A software tool or hardware device that runs concurrently with the component or system under test and supervises, records and/or analyzes the behavior of the component or system.
Reference: ISO 24765

multiplayer testing: Testing to determine if many players can simultaneously interact with the casino game world, with computer-controlled opponents, game servers, and with each other, as expected according to the game design.

multiple condition coverage: The coverage of all possible combinations of all single condition outcomes within one statement.

multiple condition testing: A white-box test technique in which test cases are designed to exercise outcome combinations of atomic conditions.

Myers-Briggs Type Indicator: (MBTI) An indicator of psychological preference representing the different personalities and communication styles of people.

negative testing: Testing a component or system in a way for which it was not intended to be used.

neighborhood integration testing: A type of integration testing in which all of the nodes that connect to a given node are the basis for the integration testing.

network zone: A sub-network with a defined level of trust. For example, the Internet or a public zone would be considered to be untrusted.

neuron coverage: The coverage of activated neurons in the neural network for a set of tests.

non-functional testing: Testing performed to evaluate that a component or system complies with non-functional requirements.

non-repudiation: The degree to which actions or events can be proven to have taken place, so that the actions or events cannot be repudiated later.
Reference: After ISO 25010

N-switch coverage: The coverage of sequences of N+1 transitions.
Reference: Chow

offline MBT: Model-based test approach whereby test cases are generated into a repository for future execution.

online MBT: Model-based test approach whereby test cases are generated and executed simultaneously.

open–loop-system: A system in which controlling action or input is independent of the output or changes in output.
Reference: Bakshi

open-source tool: A software tool that is available to all potential users in source code form, usually via the internet. Its users are permitted, usually under license, to study, change, improve and, at times, to distribute the software.

operability: The degree to which a component or system has attributes that make it easy to operate and control.
Reference: After ISO 25010

operational acceptance testing: A type of acceptance testing performed to determine if operations and/or systems administration staff can accept a system.

operational profile: An actual or predicted pattern of use of the component or system.

operational profiling: The process of developing and implementing an operational profile.
See also: operational profile

organizational test strategy: A strategy that describes the generic requirements for testing and how to perform testing within an organization.

outsourced testing: Testing performed by people who are not co-located with the project team and are not fellow employees.

pair testing: A test approach in which two team members simultaneously collaborate on testing a work product.

pairwise integration testing: A type of integration testing that targets pairs of components that work together as shown in a call graph.

pairwise testing: A black-box test technique in which test cases are designed to exercise pairs of parameter-value pairs.
Reference: After ISO 29119-4

par sheet testing: Testing to determine that the game returns the correct mathematical results to the screen, to the players' accounts, and to the casino account.

pass/fail criteria: Decision rules used to determine whether a test item has passed or failed.
Reference: After ISO 29119-1

passed: The status of a test result if the actual result matches the expected result.

password cracking: A security attack recovering secret passwords stored in a computer system or transmitted over a network.
Reference: after NIST.IR.7298

path: A sequence of consecutive edges in a directed graph.

path testing: A white-box test technique in which test cases are designed to execute paths in a control flow graph.

peer review: A review performed by others with the same abilities to create the work product.
Reference: After ISO 20246

penetration testing: A testing technique aiming to exploit security vulnerabilities (known or unknown) to gain unauthorized access.

performance efficiency: The degree to which a component or system uses time, resources and capacity when accomplishing its designated functions.
Reference: After ISO 25010

performance testing: Testing to determine the performance efficiency of a component or system.

performance testing tool: A test tool that generates load for a designated test item and that measures and records its performance during test execution.

perspective-based reading: A review technique in which a work product is evaluated from the perspective of different stakeholders with the purpose to derive other work products.

pharming: A security attack intended to redirect a website's traffic to a fraudulent website without the user's knowledge or consent.

phase containment: The percentage of defects that are removed in the same phase of the software lifecycle in which they were introduced.

phishing: An attempt to acquire personal or sensitive information by masquerading as a trustworthy entity in an electronic communication.

planning poker: A consensus-based estimation technique, mostly used to estimate effort or relative size of user stories in Agile software development. It is a variation of the Wideband Delphi method using a deck of cards with values representing the units in which the team estimates.
Reference: Mountain Goat Software

player perspective testing: Testing done by testers from a player's perspective to validate player satisfaction.

playtest: Ad hoc testing of a game by players to identify failures and gather feedback.

portability: The degree to which a component or system can be transferred from one hardware, software or other operational or usage environment to another.
Reference: After ISO 25010

postcondition: The expected state of a test item and its environment at the end of test case execution.

post-release testing: A type of testing to ensure that the release is performed correctly and the application can be deployed.

precondition: The required state of a test item and its environment prior to test case execution.

priority: The level of (business) importance assigned to an item, e.g., defect.

PRISMA: A systematic approach to risk-based testing that employs product risk identification and analysis to create a product risk matrix based on likelihood and impact. Term is derived from Product RISk MAnagement.

probe effect: An unintended change in behavior of a component or system caused by measuring it.

process assessment: A disciplined evaluation of an organization's software processes against a reference model.
Reference: after ISO 15504

process-compliant test strategy: A test strategy whereby the test team follows a set of predefined processes, whereby the processes address such items as documentation, the proper identification and use of the test basis and test oracle(s), and the organization of the test team.

process-driven scripting: A scripting technique where scripts are structured into scenarios which represent use cases of the software under test. The scripts can be parameterized with test data.

product risk: A risk that impacts the quality of a product.
See also: risk

product-based quality: A view of quality measured by the degree that well-defined quality characteristics are met.

project risk: A risk that impacts project success.
See also: risk

proximity-based testing: A type of testing to confirm that sensors can detect nearby objects without physical contact.

pseudo-oracle: An independently derived variant of the test item used to generate results, which are compared with the results of the original test item based on the same test inputs.
Reference: ISO 29119-11

quality: The degree to which a component or system satisfies the stated and implied needs of its stakeholders.
Reference: After IREB

quality assistance: An approach to quality management that focuses on a quality culture throughout an organization.

quality assurance: (QA) Activities focused on providing confidence that quality requirements will be fulfilled.
Reference: After ISO 24765
See also: quality management

quality characteristic: A category of quality attributes that bears on work product quality.
Reference: ISO 24765

quality coaching: The activities focused on helping an agile organization identify, understand, and deal with quality management, business value, flow of work, and customer collaboration.

quality control: (QC) Activities designed to evaluate the quality of a component or system.
Reference: after ISO 24765
See also: testing

quality culture: An organizational value system that results in an environment to establish and continually improve quality.

quality debt: The implied cost of deferred quality assurance activities.

quality function deployment: (QFD) A facilitated workshop technique that helps determine critical characteristics for new product development.
Reference: ISO 24765

quality management: The process of establishing and directing a quality policy, quality objectives, quality planning, quality control, quality assurance, and quality improvement for an organization.
Reference: After ISO 24765

quality risk: A product risk related to a quality characteristic.
See also: quality characteristic, product risk

RACI matrix: A matrix describing the participation by various roles in completing tasks or deliverables for a project or process. It is especially useful in clarifying roles and responsibilities. RACI is an acronym derived from the four key responsibilities most typically used: Responsible, Accountable, Consulted, and Informed.

ramp-down: A technique for decreasing the load on a system in a measurable and controlled way.

ramp-up: A technique for increasing the load on a system in a measurable and controlled way.

random testing: A black-box test technique in which input values are randomly generated based on an operational profile.

reactive test strategy: A test strategy whereby the test team waits to design and implement tests until the software is received, reacting to the actual system under test.

reactive testing: Testing that dynamically responds to the behavior of the test object and to test results being obtained.

reconnaissance: The exploration of a target area aiming to gain information that can be useful for an attack.

recoverability: The degree to which a component or system can recover the data directly affected by an interruption or a failure and re-establish the desired state of the component or system.
Reference: After ISO 25010

regression testing: A type of change-related testing to detect whether defects have been introduced or uncovered in unchanged areas of the software.

regression-averse test strategy: A test strategy whereby the test team applies various techniques to manage the risk of regression such as functional and/or non-functional regression test automation at one or more levels.

regulatory acceptance testing: A type of acceptance testing performed to verify whether a system conforms to relevant laws, policies and regulations.

reliability: The degree to which a component or system performs specified functions under specified conditions for a specified period of time.
Reference: After ISO 25010

reliability growth model: A model that shows the growth in reliability over time of a component or system as a result of the defect removal.

remote test lab: A facility that provides remote access to a test environment.

replaceability: The degree to which a component or system can replace another specified component or system for the same purpose in the same environment.
Reference: After ISO 25010

requirement: A provision that contains criteria to be fulfilled.
Reference: ISO 24765

requirements-based testing: An approach to testing in which test conditions are based on requirements.

resource utilization: The degree to which the amounts and types of resources used by a component or system, when performing its functions, meet requirements.
Reference: After ISO 25010

retrospective: A regular event in which team members discuss results, review their practices, and identify ways to improve.

reusability: The degree to which a work product can be used in more than one system, or in building other work products.
Reference: After ISO 25010

review: A type of static testing in which a work product or process is evaluated by one or more individuals to detect defects or to provide improvements.

review plan: A document describing the approach, resources and schedule of intended review activities. It identifies, amongst others: documents and code to be reviewed, review types to be used, participants, as well as entry and exit criteria to be applied in case of formal reviews, and the rationale for their choice. It is a record of the review planning process.

reviewer: A participant in a review who identifies defects in the work product.
Reference: After ISO 20246

risk: A factor that could result in future negative consequences.

risk analysis: The overall process of risk identification and risk assessment.

risk assessment: The process to examine identified risks and determine the risk level.

risk control: The overall process of risk mitigation and risk monitoring.

risk identification: The process of finding, recognizing and describing risks.
Reference: ISO 31000

risk impact: The damage that will be caused if the risk becomes an actual outcome or event.

risk level: The measure of a risk defined by impact and likelihood.
Synonym: risk exposure

risk likelihood: The probability that a risk will become an actual outcome or event.

risk management: The process for handling risks.
Reference: After ISO 24765

risk mitigation: The process through which decisions are reached and protective measures are implemented for reducing or maintaining risks to specified levels.

risk monitoring: The activity that checks and reports the status of known risks to stakeholders.

risk-based testing: Testing in which the management, selection, prioritization, and use of testing activities and resources are based on corresponding risk types and risk levels.
Reference: After ISO 29119-1

role-based review: A review technique in which a work product is evaluated from the perspective of different stakeholders.

root cause: A source of a defect such that if it is removed, the occurrence of the defect type is decreased or removed.
Reference: CMMI

root cause analysis: An analysis technique aimed at identifying the root causes of defects.

S.M.A.R.T. goal methodology: (SMART) A methodology whereby objectives are defined very specifically rather than generically. SMART is an acronym derived from the attributes of the objective to be defined: Specific, Measurable, Attainable, Relevant and Timely.

safety integrity level: (SIL) The level of risk reduction provided by a safety function, related to the frequency and severity of perceived hazards.
Reference: After IEC 61508

salting: A cryptographic technique that adds random data (salt) to the user data prior to hashing.
See also: hashing

scalability: The degree to which a component or system can be adjusted for changing capacity.
Reference: After Gerrard

scalability testing: Testing to determine the scalability of the software product.

scenario-based review: A review technique in which a work product is evaluated to determine its ability to address specific scenarios.

scribe: A person who records information at a review meeting.
Reference: After ISO 24765

script kiddie: A person who executes security attacks that have been created by other hackers rather than creating one's own attacks.
See also: hacker

scripted testing: Testing (manual or automated) that follows a test script.

security: The degree to which a component or system protects its data and resources against unauthorized access or use and secures unobstructed access and use for its legitimate users.
Reference: After ISO 25010

security attack: An attempt to gain unauthorized access to a component or system, resources, information, or an attempt to compromise system integrity.
Reference: after NIST.IR.7298

security audit: An audit evaluating an organization's security processes and infrastructure.

security policy: A high-level document describing the principles, approach and major objectives of the organization regarding security.

security procedure: A set of steps required to implement the security policy and the steps to be taken in response to a security incident.

security risk: A quality risk related to security.

security testing: Testing to determine the security of the software product.

security vulnerability: A weakness in the system that could allow for a successful security attack.

sequential development model: A type of software development lifecycle model in which a complete system is developed in a linear way of several discrete and successive phases with no overlap between them.

service virtualization: A technique to enable virtual delivery of services which are deployed, accessed and managed remotely.

session-based test management: (SBTM) A method for measuring and managing session-based testing.

session-based testing: A test approach in which test activities are planned as test sessions.

severity: The degree of impact that a defect has on the development or operation of a component or system.

shift left: An approach to performing testing and quality assurance activities as early as possible in the software development lifecycle.

short-circuiting: A programming language/interpreter technique for evaluating compound conditions in which a condition on one side of a logical operator may not be evaluated if the condition on the other side is sufficient to determine the final outcome.

sign change coverage: The coverage of neurons activated with both positive and negative activation values in a neural network for a set of tests.

sign-sign coverage: The coverage achieved if by changing the sign of each neuron it can be shown to individually cause one neuron in the next layer to change sign while all other neurons in the next layer do not change sign for a set of tests.

simulator: A component or system used during testing which behaves or operates like a given component or system.
Reference: ISO 24765

smoke test: A test suite that covers the main functionality of a component or system to determine whether it works properly before planned testing begins.

social engineering: An attempt to trick someone into revealing information (e.g., a password) that can be used to attack systems or networks.
Reference: NIST.IR.7298

software development lifecycle: (SDLC) The activities performed at each stage in software development, and how they relate to one another logically and chronologically.

software in the loop: (SiL) Dynamic testing performed using real software in a simulated environment or with experimental hardware.
Reference: Automotive SPICE

software lifecycle: The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software lifecycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and sometimes, retirement phase. Note these phases may overlap or be performed iteratively.

software process improvement: (SPI) A program of activities designed to improve the performance and maturity of the organization's software processes and the results of such a program.
Reference: After CMMI

software qualification test: Testing performed on completed, integrated software to provide evidence for compliance with software requirements.
Reference: Automotive SPICE

Software Usability Measurement Inventory: (SUMI) A questionnaire-based usability testing tool that measures and benchmarks user experience.
Reference: Kirakowski93

source test case: A test case that passed and is used as the basis of follow-up test cases in metamorphic testing.

specification by example: (SBE) A development technique in which the specification is defined by examples.
See also: acceptance test-driven development

spike testing: Testing to determine the ability of a system to recover from sudden bursts of peak loads and return to a steady state.

SQL injection: A type of code injection in the structured query language (SQL).

standard-compliant test strategy: A test strategy whereby the test team follows a standard. Standards followed may be valid e.g., for a country (legislation standards), a business domain (domain standards), or internally (organizational standards).

state transition testing: A black-box test technique in which test cases are designed to exercise elements of a state transition model.

statement coverage: The coverage of executable statements.

statement testing: A white-box test technique in which test cases are designed to execute statements.

static analysis: The process of evaluating a component or system without executing it, based on its form, structure, content, or documentation.
Reference: After ISO 24765

static testing: Testing that does not involve the execution of a test item.

stress testing: A type of performance testing conducted to evaluate a system or component at or beyond the limits of its anticipated or specified workloads, or with reduced availability of resources such as access to memory or servers.
Reference: ISO 24765

structural coverage: Coverage measures based on the internal structure of a component or system.

structured scripting: A scripting technique that builds and utilizes a library of reusable (parts of) scripts.

stub: A skeletal or special-purpose implementation of a software component, used to develop or test a component that calls or is otherwise dependent on it. It replaces a called component.
Reference: After ISO 24765

summative evaluation: A type of evaluation designed and used to gather conclusions about the quality of a component or system, especially when a substantial part of it has completed design.
See also: formative evaluation, testing

system hardening: The step-by-step process of reducing the security vulnerabilities of a system by applying a security policy and different layers of protection.

system integration testing: A test level that focuses on interactions between systems.

system of systems: Multiple heterogeneous, distributed systems that are embedded in networks at multiple levels and in multiple interconnected domains, addressing large-scale inter-disciplinary common problems and purposes, usually without a common management structure.

system qualification test: Testing performed on the completed, integrated system of software components, hardware components, and mechanics to provide evidence for compliance with system requirements and that the complete system is ready for delivery.
Reference: Automotive SPICE

system testing: A test level that focuses on verifying that a system as a whole meets specified requirements.

system throughput: The amount of data passing through a component or system in a given time period.
Reference: After ISO 24765

system under test: (SUT) A type of test object that is a system.

System Usability Scale: (SUS) A simple, ten-item attitude scale giving a global view of subjective assessments of usability.

Systematic Test and Evaluation Process: (STEP) A structured testing methodology also used as a content-based model for improving the testing process. It does not require that improvements occur in a specific order.

technical review: A formal review by technical experts that examine the quality of a work product and identify discrepancies from specifications and standards.

test: A set of one or more test cases.

test adaptation layer: The layer in a test automation architecture which provides the necessary code to adapt test scripts on an abstract level to the various components, configuration or interfaces of the SUT.

test analysis: The activity that identifies test conditions by analyzing the test basis.

test approach: The manner of implementing testing tasks.

test architect: (1) A person who provides guidance and strategic direction for a test organization and for its relationship with other disciplines. (2) A person who defines the way testing is structured for a given system, including topics such as test tools and test data management.

test automation: The use of software to perform or support test activities.

test automation architecture: An instantiation of the generic test automation architecture to define the architecture of a test automation solution, i.e., its layers, components, services and interfaces.

test automation engineer: A person who is responsible for the design, implementation and maintenance of a test automation architecture as well as the technical evolution of the resulting test automation solution.

test automation framework: A tool that provides an environment for test automation. It usually includes a test harness and test libraries.

test automation manager: A person who is responsible for the planning and supervision of the development and evolution of a test automation solution.

test automation solution: A realization/implementation of a test automation architecture, i.e., a combination of components implementing a specific test automation assignment. The components may include commercial off-the-shelf test tools, test automation frameworks, as well as test hardware.

test automation strategy: A high-level plan to achieve long-term objectives of test automation under given boundary conditions.

test basis: The body of knowledge used as the basis for test analysis and design.
Reference: After TMap

test case: A set of preconditions, inputs, actions (where applicable), expected results and postconditions, developed based on test conditions.

test case explosion: The disproportionate growth of the number of test cases with growing size of the test basis, when using a certain test design technique. Test case explosion may also happen when applying the test design technique systematically for the first time.

test charter: Documentation of the goal or objective for a test session.

test closure: During the test closure phase of a test process data is collected from completed activities to consolidate experience, testware, facts and numbers. The test closure phase consists of finalizing and archiving the testware and evaluating the test process, including preparation of a test evaluation report.

test completion: The activity that makes testware available for later use, leaves test environments in a satisfactory condition and communicates the results of testing to relevant stakeholders.

test completion report: A type of test report produced at completion milestones that provides an evaluation of the corresponding test items against exit criteria.

test condition: A testable aspect of a component or system identified as a basis for testing.
Reference: After ISO 29119-1

test control: The activity that develops and applies corrective actions to get a test project on track when it deviates from what was planned.

test cycle: An instance of the test process against a single identifiable version of the test object.

test data: Data needed for test execution.

test data preparation: The activity to select data from existing databases or create, generate, manipulate and edit data for testing.

test definition layer: The layer in a generic test automation architecture which supports test implementation by supporting the definition of test suites and/or test cases, e.g., by offering templates or guidelines.

test design: The activity that derives and specifies test cases from test conditions.

test director: A senior manager who manages test managers.
See also: test manager

test environment: An environment containing hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.
Reference: ISO 24765

test estimation: An approximation related to various aspects of testing.

test execution: The activity that runs a test on a component or system producing actual results.

test execution automation: The use of software, e.g., capture/playback tools, to control the execution of tests, the comparison of actual results to expected results, the setting up of test preconditions, and other test control and reporting functions.

test execution layer: The layer in a generic test automation architecture which supports the execution of test suites and/or test cases.

test execution schedule: A schedule for the execution of test suites within a test cycle.

test execution tool: A test tool that executes tests against a designated test item and evaluates the outcomes against expected results and postconditions.

test generation layer: The layer in a generic test automation architecture which supports manual or automated design of test suites and/or test cases.

test harness: A collection of stubs and drivers needed to execute a test suite

test hook: A customized software interface that enables automated testing of a test object.

test implementation: The activity that prepares the testware needed for test execution based on test analysis and design.

test improvement plan: A plan for achieving organizational test process improvement objectives based on a thorough understanding of the current strengths and weaknesses of the organization's test processes and test process assets.
Reference: After CMMI

test infrastructure: The artifacts needed to perform testing, consisting of test environments, test tools, office environment and procedures.

test item: A part of a test object used in the test process.
See also: test object

test leader: On large projects, the person who reports to the test manager and is responsible for project management of a particular test level or a particular set of testing activities.
See also: test manager

test level: A specific instantiation of a test process.
Reference: ISO 29119-1

test log: A chronological record of relevant details about the execution of tests.
Reference: ISO 24765

test logging: The activity of creating a test log.

test management: The process of planning, scheduling, estimating, monitoring, reporting, controlling, and completing test activities.
Reference: ISO 29119-1

test management tool: A tool that supports test management.

test manager: The person responsible for project management of testing activities, resources, and evaluation of a test object.

Test Maturity Model integration: (TMMi) A five-level staged framework for test process improvement, related to the Capability Maturity Model Integration (CMMI), that describes the key elements of an effective test process.

test mission: The purpose of testing for an organization, often documented as part of the test policy.
See also: test policy

test model: A model describing testware that is used for testing a component or a system under test.

test monitoring: The activity that checks the status of testing activities, identifies any variances from planned or expected, and reports status to stakeholders.

test object: The work product to be tested.

test objective: The purpose for testing.

test oracle: A source to determine an expected result to compare with the actual result of the system under test.
Reference: After Adrion

test plan: Documentation describing the test objectives to be achieved and the means and the schedule for achieving them, organized to coordinate testing activities.
Reference: After ISO 29119-1

test planning: The activity of establishing or updating a test plan.

test point analysis: (TPA) A formula-based test estimation technique based on function point analysis.
Reference: TMap

test policy: A high-level document describing the principles, approach and major objectives of the organization regarding testing.

test procedure: A sequence of test cases in execution order, and any associated actions that may be required to set up the initial preconditions and any wrap up activities post execution.
Reference: ISO 29119-1

test process: The set of interrelated activities comprising of test planning, test monitoring and control, test analysis, test design, test implementation, test execution, and test completion.

test process group: (TPG) A collection of specialists who facilitate the definition, maintenance, and improvement of the test processes used by an organization.
Reference: After CMMI

test process improvement: A program of activities undertaken to improve the performance and maturity of the organization's test processes.
Reference: After CMMI

test process improvement manifesto: A statement that echoes the Agile manifesto, and defines values for improving the test process.
Reference: Veenendaal08

test process improver: A person implementing improvements in the test process based on a test improvement plan.

test progress report: A type of test report produced at regular intervals about the progress of test activities against a baseline, risks, and alternatives requiring a decision.

test pyramid: A graphical model representing the relationship of the amount of testing per level, with more at the bottom than at the top.

test report: Documentation summarizing test activities and results.

test reporting: Collecting and analyzing data from testing activities and subsequently consolidating the data in a report to inform stakeholders.

test result: The consequence/outcome of the execution of a test.

test run: The execution of a test suite on a specific version of the test object.

test scope: A description of the test object and its features to be tested.

test script: A sequence of instructions for the execution of a test.

test selection criteria: The criteria used to guide the generation of test cases or to select test cases in order to limit the size of a test.

test session: An uninterrupted period of time spent in executing tests.

test specification: The complete documentation of the test design, test cases, and test scripts for a specific test item.
Reference: After ISO 29119-1

test step: A single interaction between an actor and the test object consisting of an input, an action, and an expected result.

test strategy: Documentation aligned with the test policy that describes the generic requirements for testing and details how to perform testing within an organization.

test suite: A set of test scripts or test procedures to be executed in a specific test run.

test technique: A procedure used to define test conditions, design test cases, and specify test data.

test type: A group of test activities based on specific test objectives aimed at specific characteristics of a component or system.
Reference: After TMap

testability: The degree to which test conditions can be established for a component or system, and tests can be performed to determine whether those test conditions have been met.
Reference: After ISO 25010

test-driven development: (TDD) A software development technique in which the test cases are developed, automated and then the software is developed incrementally to pass those test cases.

tester: A person who performs testing.

test-first approach: An approach to software development in which the test cases are designed and implemented before the associated component or system is developed.
See also: test-driven development

testing: The process within the software development lifecycle that evaluates the quality of a component or system and related work products.

testing quadrants: A classification model of test types/test levels in four quadrants, relating them to two dimensions of test objectives: supporting the product team versus critiquing the product, and technology-facing versus business-facing.

testware: Work products produced during the test process for use in planning, designing, executing, evaluating and reporting on testing.
Reference: After ISO 29119-1

think aloud usability testing: A usability testing technique where test participants share their thoughts with the moderator and observers by thinking aloud while they solve usability test tasks. Think aloud is useful to understand the test participant.

think time: The amount of time required by a user to determine and execute the next action in a sequence of actions.

threshold coverage: The coverage of neurons exceeding a threshold activation value in a neural network for a set of tests.

time behavior: The degree to which a component or system can perform its required functions within required response times, processing times and throughput rates.
Reference: After ISO 25010

Total Quality Management: (TQM) An organization-wide management approach to quality based on employee participation to achieve long-term success through customer satisfaction.
Reference: After ISO 24765

tour: A set of exploratory tests organized around a special focus.

TPI Next: A continuous business-driven framework for test process improvement that describes the key elements of an effective and efficient test process.

traceability: The ability to establish explicit relationships between related work products or items within work products.
Reference: After ISO 19506

traceability matrix: A two-dimensional table, which correlates two entities (e.g., requirements and test cases). The table allows tracing back and forth the links of one entity to the other, thus enabling the determination of coverage achieved and the assessment of impact of proposed changes.

transcendent quality: A view of quality based on the perception and feeling of individuals.
Reference: After Garvin

unit test framework: A tool that provides an environment for unit or component testing in which a component can be tested in isolation or with suitable stubs and drivers. It also provides other support for the developer, such as debugging capabilities.
Reference: Graham

usability: The degree to which a component or system can be used by specified users to achieve specified goals in a specified context of use.
Reference: After ISO 25010

usability evaluation: A process through which information about the usability of a system is gathered in order to improve the system (known as formative evaluation) or to assess the merit or worth of a system (known as summative evaluation).
See also: formative evaluation, summative evaluation

usability lab: A test facility in which unintrusive observation of participant reactions and responses to software takes place.

usability requirement: A requirement on the usability of a component or system.

usability test participant: A representative user who solves typical tasks in a usability test.

usability test script: A document specifying a sequence of actions for the execution of a usability test. It is used by the moderator to keep track of briefing and pre-session interview questions, usability test tasks, and post-session interview questions.

usability test session: A test session in usability testing in which a usability test participant is executing tests, moderated by a moderator and observed by a number of observers.

usability test task: A usability test execution activity specified by the moderator that needs to be accomplished by a usability test participant within a given period of time.

usability testing: Testing to evaluate the degree to which the system can be used by specified users with effectiveness, efficiency and satisfaction in a specified context of use.
Reference: After ISO 25010

use case testing: A black-box test technique in which test cases are designed to exercise use case behaviors.

user acceptance testing: (UAT) A type of acceptance testing performed to determine if intended users accept the system.
See also: acceptance testing

user error protection: The degree to which a component or system protects users against making errors.
Reference: After ISO 25010

user experience: A person's perceptions and responses resulting from the use or anticipated use of a software product.
Reference: ISO 9241-210

user interface: All components of a system that provide information and controls for the user to accomplish specific tasks with the system.

user interface aesthetics: The degree to which a user interface enables pleasing and satisfying interaction for the user.
Reference: ISO 25010

user interface guideline: A low-level, specific rule or recommendation for user interface design that leaves little room for interpretation so designers implement it similarly. It is often used to ensure consistency in the appearance and behavior of the user interface of the systems produced by an organization.

user story: A user or business requirement consisting of one sentence expressed in the everyday or business language which is capturing the functionality a user needs, the reason behind it, any non-functional criteria, and also including acceptance criteria.

user story testing: A black-box test technique in which test conditions are the acceptance criteria of user stories.

user survey: A usability evaluation whereby a representative sample of users are asked to report subjective evaluation into a questionnaire based on their experience in using a component or system.

user-agent based testing: A type of testing in which a test client is used to switch the user agent string and identify itself as a different client while executing test suites.

user-based quality: A view of quality measured by the degree that the needs, wants, and desires of a user are met.
Reference: after Garvin
See also: manufacturing-based quality, product-based quality, transcendent-based quality, value-based quality

validation: Confirmation by examination that a work product matches a stakeholder's needs.
Reference: ISO 9000

value change coverage: The coverage of neurons activated where their activation values differ by more than a change amount in the neural network for a set of tests.

value-based quality: A view of quality measured by the ratio of the cost to the value received from a product or service.
Reference: After Garvin
See also: manufacturing-based quality, product-based quality, transcendent-based quality, user-based quality

verification: Confirmation by examination and through provision of objective evidence that specified requirements have been fulfilled.
Reference: ISO 9000

virtual test environment: A test environment in which one or more parts are digitally simulated.
Reference: ISO 29119-11

virtual user: A simulation of activities performed according to a user operational profile.

visual testing: Testing that uses image recognition to interact with GUI objects.
Reference: After ISO 29119-11

V-model: A sequential development lifecycle model describing a one-for-one relationship between major phases of software development from business requirements specification to delivery, and corresponding test levels from acceptance testing to component testing.

vulnerability scanner: A static analyzer that is used to detect particular security vulnerabilities in the code.

walkthrough: A type of review in which an author leads members of the review through a work product and the members ask questions and make comments about possible issues.
Reference: After ISO 20246

Web Content Accessibility Guidelines: (WCAG) A part of a series of web accessibility guidelines published by the Web Accessibility Initiative (WAI) of the World Wide Web Consortium (W3C), the main international standards organization for the internet. They consist of a set of guidelines for making content accessible, primarily for people with disabilities.

Website Analysis and Measurement Inventory: (WAMMI) A commercial website analysis service providing a questionnaire for measuring user experience and assessing delivery of business goals online.

white-box test technique: A test technique only based on the internal structure of a component or system.

white-box testing: Testing based on an analysis of the internal structure of the component or system.

Wideband Delphi: An expert-based test estimation technique that aims at making an accurate estimation using the collective wisdom of the team members.

wild pointer: A pointer that references a location that is out of scope for that pointer or that does not exist.

XiL test environment: (XiL) A generalized term for dynamic testing in different virtual test environments.

Appendix A

Standards

[BS 7925/2] BS 2925:2001, Software Component Testing Standard, BCS SIGIST Working Draft 3.4

[DO-178b] DO-178B:1992. Software Considerations in Airborne Systems and Equipment Certification, Requirements and Technical Concepts for Aviation (RTCA SC167)

[IEEE 610] IEEE 610.12:1990. Standard Glossary of Software Engineering Terminology.

[IEEE 730] IEEE 730:2002. Software Quality Assurance Plans

[IEEE 829] IEEE 829:1998. Standard for Software Test Documentation

[IEEE 1008] IEEE 1008:1993. Standard for Software Unit Testing

[IEEE 1028] IEEE 1028:1997. Standard for Software Reviews and Audits

[IEEE 1044] IEEE 1044:1993. Standard Classification for Software Anomalies

[IEEE 1219] IEEE 1219:1998. Software Maintenance

[ISO 2382] ISO/IEC 2382-1:1993. Data processing - Vocabulary - Part 1: Fundamental terms

[ISO 8402] ISO 8402: 1994. Quality Management and Quality Assurance Vocabulary

[ISO 9000] ISO 9000:2005. Quality Management Systems – Fundamentals and Vocabulary

[ISO 9126] ISO/IEC 9126-1:2001. Software Engineering – Software Product Quality – Part 1: Quality characteristics and sub-characteristics

[ISO 9241] ISO 9241:2010. Ergonomics of human-system interaction – Part 210: Human-centered design for interactive systems

[ISO 12207] ISO/IEC 12207:1995. Information Technology – Software Lifecycle Processes

[ISO 14598] ISO/IEC 14598-1:1999. Information Technology – Software Product Evaluation - Part 1: General Overview

[ISO 14764] ISO/IEC 14764:2006. Software Engineering - Software Life Cycle Processes - Maintenance

[ISO 15504] ISO/IEC 15504-9: 1998. Information Technology – Software Process Assessment – Part 9: Vocabulary

[ISO 19506] ISO/IEC 19506:2012. Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM)

[ISO 20246] ISO/IEC 20246:2017. Software and systems engineering -- Work product reviews

[ISO 24765] ISO/IEC/IEEE 24765:2017. Systems and software engineering -- Vocabulary

[ISO 25010] ISO/IEC 25010:2011. Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models

[ISO 25040] ISO/IEC 25040:2011. Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- Evaluation process

[ISO 29119] ISO/IEC/IEEE 29119-1:2013. Software and systems engineering -- Software testing -- Part 1: Concepts and definitions

[ISO 31000] ISO 31000:2018. Risk management

[NIST.IR.7298] U.S. Department of Commerce, National Institute of Standards and Technology – Glossary of Key Information Security Terms, Revision 2, May 2013

Books and papers

[Adrion] W. Adrion, M. Branstad and J. Cherniabsky (1982), Validation, Verification and Testing of Computer Software, in: Computing Surveys, Vol. 14, No 2, June 1982

[Akao] Akao, Yoji (1994), Development History of Quality Function Deployment - The Customer Driven Approach to Quality Planning and Deployment, Minato, Tokyo 107 Japan: Asian Productivity Organization, pp. 339, ISBN 92-833-1121-3

[Bach] J. Bach (2004), Exploratory Testing, in: E. van Veenendaal, The Testing Practitioner – 2nd edition, UTN Publishing, ISBN 90-72194-65-9

[Beizer] B. Beizer (1990), Software Testing Techniques, van Nostrand Reinhold, ISBN 0-442-20672-0

[Chow] T. Chow (1978), Testing Software Design Modelled by Finite-Sate Machines, in: IEEE Transactions on Software Engineering, Vol. 4, No 3, May 1978

[CMMI] M.B. Chrissis, M. Konrad and S. Shrum (2004), CMMi, Guidelines for Process Integration and Product Improvement, Addison Wesley, ISBN 0-321-15496-7

[Deming] D. W. Edwards (1986), Out of the Crisis, MIT Center for Advanced Engineering Study, ISBN 0-911379-01-0

[Egler63] J. F. Egler. 1963. A procedure for converting logic table conditions into an efficient sequence of test instructions. Commun. ACM 6, 9 (September 1963), 510-514. DOI=10.1145/367593.367595

[Fenton] N. Fenton (1991), Software Metrics: a Rigorous Approach, Chapman & Hall, ISBN 0-53249-425-1

[Fewster and Graham] M. Fewster and D. Graham (1999), Software Test Automation, Effective use of test execution tools, Addison-Wesley, ISBN 0-201-33140-3

[Freedman and Weinberg] D. Freedman and G. Weinberg (1990), Walkthroughs, Inspections, and Technical Reviews, Dorset House Publishing, ISBN 0-932633-19-6

[Garvin] D.A. Garvin (1984), What does product quality really mean?, in: Sloan Management Review, Vol. 26, nr. 1 1984

[Gerrard] P. Gerrard and N. Thompson (2002), Risk-Based E-Business Testing, Artech House Publishers, ISBN 1-58053-314-0

[Gilb and Graham] T. Gilb and D. Graham (1993), Software Inspection, Addison-Wesley, ISBN 0-201-63181-4

[Graham] D. Graham, E. van Veenendaal, I. Evans and R. Black (2007), Foundations of Software Testing, Thomson Learning, ISBN 978-1-84480-355-2

[Grochtmann] M. Grochtmann (1994), Test Case Design Using Classification Trees, in: Conference Proceedings STAR 1994

[Hetzel] W. Hetzel (1988), The complete guide to software testing – 2nd edition, QED Information Sciences, ISBN 0-89435-242-3

[Juran] J.M. Juran (1979), Quality Control Handbook, McGraw-Hill

[Kirakowski93] J. Kirakowski, M Corbett (1993), SUMI: the Software Usability Measurement Inventory, British Journal of Educational Technology, Volume 24, Issue 3, pages 210–212, September 1993

[McCabe] T. McCabe (1976), A complexity measure, in: IEEE Transactions on Software Engineering, Vol. 2, pp. 308-320

[Musa] J. Musa (1998), Software Reliability Engineering Testing, McGraw-Hill Education, ISBN 0-07913-271-5

[PMBOK]

[TMap] M. Pol, R. Teunissen, E. van Veenendaal (2002), Software Testing, A guide to the TMap Approach, Addison Wesley, ISBN 0-201-745712

[TMMi] E. van Veenendaal and J. Cannegieter (2011), The Little TMMi, UTN Publishing, ISBN 97-89490986-03-2

[Veenendaal08] E. van Veenendaal (2008), Test Improvement Manifesto, in: Testing Experience, Issue 04/08, December 2008

Internet

[extremeprogramming.org] http://www.extremeprogramming.org/; retrieved on the 04th of June, 2018.

Trademarks

In the ISTQB Glossary the following trademarks are used:
- CMMi and IDEAL are registered trademarks of Carnegie Mellon University
- EFQM is a registered trademark of the EFQM Foundation
- Rational Unified Process (RUP) is a registered trademark of Rational Software Corporation
- STEP is a registered trademark of Software Quality Engineering
- TMap, TPA and TPI Next are registered trademarks of Sogeti Nederland BV
- TMMi is a registered trademark of the TMMi Foundation

Appendix B

Suggestions and comments for improvements to this glossary are warmly welcomed.

When submitting suggestions and comments do not forget to enter the following information:
– your name and contact options,
– version number of the glossary the comment applies
– detailed information about which part of the glossary the comment applies to and
– information on proposed change and a brief explanation for the amendment to be introduced.

Suggestions and comments are sent to *info@sstb.se*.