


| | | |
|---|--|--|
|  | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 Page 1 of 26 |

R214 – SPECIFIC REQUIREMENTS:
INFORMATION TECHNOLOGY TESTING
LABORATORY ACCREDITATION PROGRAM

June 2010


© 2010 by A2LA All rights reserved. No part of this document may be reproduced in any form or by any means without the prior written permission of A2LA.



R214: Specific Requirements: Information Technology Testing Laboratory Accreditation
Program

Table of Contents

| | |
|---|----------|
| 1. SCOPE | 3 |
| 2. REFERENCES | 3 |
| 3. DEFINITIONS..... | 3 |
| 4. Management Requirements | 5 |
| 4.1. Organization..... | 5 |
| 4.2. Quality system | 5 |
| 4.3. Document control..... | 5 |
| 4.4. Review of requests, tenders and contracts | 5 |
| 4.5. Subcontracting of tests and calibrations | 6 |
| 4.6. Purchasing services and supplies | 6 |
| 4.7. Service to the client..... | 6 |
| 4.8. Complaints | 6 |
| 4.9. Control of nonconforming testing and/or calibration work | 6 |
| 4.10. Improvement | 6 |
| 4.11. Corrective action..... | 6 |
| 4.12. Preventive action..... | 6 |
| 4.13. Control of records | 6 |
| 4.14. Internal audits..... | 6 |
| 4.15. Management reviews | 7 |
| 5. Technical requirements | 7 |
| 5.1. General..... | 7 |
| 5.2. Personnel..... | 7 |
| 5.3. Accommodation and environmental conditions | 7 |
| 5.4. Test and calibration methods and method validation | 7 |
| 5.5. Equipment..... | 9 |
| 5.6. Measurement Traceability | 10 |
| 5.7. Sampling | 10 |
| 5.8. Handling of test and calibration items | 10 |
| 5.9. Assuring the quality of test and calibration results..... | 11 |
| 5.10. Reporting the results | 11 |

| | | |
|---|--|--|
|  | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 Page 3 of 26 |

INTRODUCTION

A2LA uses as the basis for all of its laboratory accreditations (of both testing and calibration laboratories) ISO/IEC 17025: 2005, “General requirements for the competence of testing and calibration laboratories”. These requirements have become the standard throughout the world, and identifying laboratories that have demonstrated compliance with these requirements has become the basis for mutual recognition agreements with accreditation systems in other countries. A2LA has also developed “Specific Criteria” which are additional requirements applicable to a certain field, a certain technology or a certain type of test/calibration. Laboratories seeking accreditation in any of these programs must also meet the requirements outlined in the Specific Criteria relevant to the program.

1. SCOPE

All laboratories seeking accreditation for Information Technology (IT) testing must meet the general requirements of ISO/IEC 17025: 2005. IT testing is defined as any aspect of a hardware and or software environment that is under test. This testing can be physical, logical, virtual, or analytical (see Appendix 1 for further description). This document describes additional accreditation requirements specifically applicable to laboratories performing testing related to information technology. The information contained in this document constitutes either additions to the general accreditation requirements or clarifications of the general requirements as they relate to IT testing.

2. REFERENCES


- 2.1.1. ISO/IEC 17000, “Conformity assessment – Vocabulary and general principles”
- 2.1.2. ISO/IEC 17025: 2005, “General requirements for the competence of testing and calibration laboratories”
- 2.1.3. IEEE STD 610-12:1990 “IEEE Standard Glossary of Software Engineering Terminology”

3. DEFINITIONS

- 3.1.1. **Acceptance Testing:** Formal testing conducted to determine whether or not a system satisfies acceptance criteria and to enable the customer to determine whether to accept the system [IEEE STD 610-12:1990].



- 3.1.2. **ASP (Application Service Provider):** Hosted software contractor. An ASP operates software at its data center, which customer's access online under a service contract. Specifically for testing this would apply to ASP services important to testing, supplied to the laboratory.
- 3.1.3. **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 compliance with specified requirements. [IEEE STD 610-12:1990]
- 3.1.4. **COTS (Commercial Off The Shelf) Software:** Code that is purchased without modification and either cannot or will not be modified by the lab. An example of this would be Microsoft Word/Excel/or dedicated instrument interface.
- 3.1.5. **Criticality or Severity:** The degree of impact that a requirement, module, fault, error, failure, or other item has on the development or operation of a system [IEEE STD 610-12:1990]
- 3.1.6. **Error or Fault:** The difference between a computed, observed, or measured valued or condition and the true, specified, or theoretically correct value or condition. [IEEE STD 610-12:1990]
- 3.1.7. **MOTS (Modified Off The Shelf) Software:** COTS software that is configured or adapted to a specific application. Examples include Lab Windows, Lab Tech Notebook, Tile EMC, generic data acquisition software, excel formulas, or MS Office macros, etc.
- 3.1.8. **Product:** Any COTS, MOTS, or custom software is considered a product.
- 3.1.9. **Software Life Cycle (SLC):** The period of time that begins when a software product is conceived and ends when the software is no longer available for use. [IEEE STD 610-12:1990]
- 3.1.10. **SUT (System Under Test):** The software product or system undergoing testing by the laboratory
- 3.1.11. **System:** A computing environment that contains both hardware and software. A collection of components organized to accomplish a specific function or set of functions [IEEE STD 610-12:1990]
- 3.1.12. **Test Cases:** A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement [IEEE STD 610-12:1990]

| | | |
|---|--|--|
|  | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 Page 5 of 26 |

- 3.1.13. **Test Environment:** An operating environment that emulates, as close as possible, the target environment of the SUT. The test environment includes hardware, operating system, and any other software products running on the same machine.
- 3.1.14. **Test Plan:** A document that describes the technical and management approach to be followed for testing a system or component [IEEE STD 610-12:1990]
- 3.1.15. **Test Specification:** A document that specifies the test inputs, execution conditions, and predicts results for an item to be tested. [IEEE STD 610-12:1990]
- 3.1.16. **Test Suite:** A collection of test cases to be executed as a logical group.
- 3.1.17. **Test Tools:** Software or hardware products that are used to facilitate the testing of the SUT.

4. Management Requirements

4.1. Organization

4.1 IT.1 The management system requirements of ISO/IEC 17025 and the additional requirements of this document apply to the laboratory's permanent facilities, testing performed at the customer's facility, and on any testing performed via a remote connection to the customer's or sub-contracted (such as an ASP) facility.

4.2. Quality system

No additions

4.3. Document control

No additions

4.4. Review of requests, tenders and contracts

4.4 IT.1 The contract shall define which components of the test environment are being supplied by the laboratory and which are being supplied by the client. This includes hardware and software. The test environment boundary interface points shall also be clearly defined.

4.4 IT.2 When ASP services are utilized for testing, it shall be agreed to in writing by the client in the contract review.



4.5. Subcontracting of tests and calibrations

No additions

4.6. Purchasing services and supplies

4.6 IT.1 For the purposes of accreditation in the IT field of testing, the requirements in this section of ISO/IEC 17025 also apply to the purchase of any ASP services that are used to conduct the testing.

4.7. Service to the client

No additions

4.8. Complaints

No additions

4.9. Control of nonconforming testing and/or calibration work

4.9 IT.1 Errors detected in the SUT do not constitute non-conforming work, but are an aspect of the overall results of the test. These errors shall be documented in the test report in accordance with ISO/IEC 17025 section 5.10.

4.9 IT.2 Any other aspect of testing not associated with results, that do not conform to the documented test methodologies (see section 5.4) shall be considered non-conforming work and subject to the requirements of this section.

4.10. Improvement

No additions

4.11. Corrective action

No additions

4.12. Preventive action


No additions

4.13. Control of records

4.13 IT.1 Technical records shall include, as far as possible, the correct and complete identification of the test environment used for the SUT; this includes complete configuration management identification for all system components (both hardware and software).

4.14. Internal audits

No Additions

| | | |
|---|--|--|
|  | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 Page 7 of 26 |

4.15. Management reviews

No Additions

5. Technical requirements

5.1. General

No additions

5.2. Personnel

No additions

5.3. Accommodation and environmental conditions

5.3 IT.1 IT testing should be separated from any design/development or production environments. There should be no other concurrent activities occurring during testing that could affect or invalidate the results.

5.3 IT.2 Any virtual environments or other special configurations shall be fully documented in the test records (as per 4.13.1 (IT) above) along with a justification as to why it is believed not to affect or invalidate the results.

5.3 IT.3 When ASP services are utilized for testing, any outside system influences that could be contributed from other ASP users shall be documented in the technical records.


5.3 IT.4 When a computing hosting center is utilized to house the lab-owned system hardware it is considered within and part of the lab environment.

5.4. Test and calibration methods and method validation

Note: Discussion of *test methods* as defined in ISO/IEC 17025 section 5.4 are referred to as *testing methodology(s)* in these program requirements.

5.4 IT.1 The lab shall define and document a testing methodology which shall address the following topics:

- (a) Test preparation and setup
- (b) Test coverage and traceability to requirements.
- (c) Assurance that test case results are not ambiguous and have single thread of execution with objective results relating to expected outcomes.

| | | |
|---|--|--|
|  | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 Page 8 of 26 |

- (d) Assurance that any automated test suites will produce valid results.
- (e) Test document approval prior to testing.
- (f) Completed test case review and approval.
- (g) Test reporting with anomaly severity classifications.
- (h) Test Candidate configuration management.

It may also include the following, subject to contract review:

- (i) Test anomaly characterization and priority.
- (j) Criteria for running partial testing or re-testing candidates.
- (k) Any other topics with agreement with the customer.

5.4 IT.2 IT Testing work shall be defined in Test Plans, Test Specifications, Test Cases, or other test suite deliverables as defined in the testing methodology. These can also be encompassed in an overall Validation Plan with matching Validation Report as defined by the methodology.

5.4 IT.3 The test suites/plans/specifications/cases shall be technically reviewed and approved prior to execution. This can be considered the validation of test method as defined in ISO/IEC 17025 clause 5.4.5. This review shall include:

- (a) Confirmation of adequate test coverage of all requirements.
- (b) Confirmation that test case results are not ambiguous and have objective pass/fail criteria.
- (c) Confirmation that any automated test suites will produce valid results.

5.4 IT.4 The concept of Measurement Uncertainty (MU) typically is not applicable as IT testing executes digital logic on a pass/fail basis. MU may be applied to IT under the following conditions:

- (a) When the SUT is performing mathematical operations or using approximations and rounding in statistical analysis, calculus, or geometry, an uncertainty may be introduced by the algorithms themselves. Where this becomes significant to the output or functioning of the SUT, MU shall be documented
- (b) For organizations performing testing on software used to support the Calibration and Measurement Capabilities (CMC) claims on a Scope of Accreditation, the software shall include all the necessary contributors to ensure that the measurement uncertainty values are calculated in accordance with the method detailed in the current version of ISO/IEC Guide 98 “Guide to the



Expression of Uncertainty in Measurement” (GUM). Validation of the software for compliance with the GUM shall be conducted in accordance with a defined procedure and records of the validation shall be retained.

(c) In addition to item (b) above, for organizations also performing testing on software used to verify that measurement quantities are within specified tolerances (for example to meet the requirements of the current version of ANSI Z540.3 section 5.3.b), validation of the software for compliance must be conducted in accordance with a defined procedure and records of the validation shall be retained.

5.4 IT.5 Computers as described in section 5.4.7 of ISO/IEC 17025 are considered test equipment and are managed per the requirements of ISO/IEC 17025 section 5.5.

5.5. Equipment

5.5 IT.1 Software test tools significant to testing are considered equipment and shall follow the appropriate ISO/IEC 17025 section 5.5 clauses.


5.5 IT.2 Software Tool validation confirms that the software tools meet the specified requirements. The software tools shall be validated, documented and include the following objective evidence:

- (a) Custom software testing tools – Full validation effort.
- (b) COTS software tools used as is – Acceptance testing for each installed instance.
- (c) MOTS software tools – Acceptance testing for each installed instance along with validation of the modification or tailoring.

5.5 IT.3 Each software test tool installation (instance) shall undergo a documented installation/operational qualification prior to use. There shall be documented evidence of the configuration and inventory of each specific installed software or system and suitable tests to confirm functionality.

5.5 IT.4 Software test tools can be installed on many systems. Each instance of test tool software shall be uniquely identified on each target environment and be under configuration management.

5.5 IT.5 The equipment records requirements in ISO/IEC 17025 are defined here as follows:

| | | |
|--|--|---|
|  <p>World Class Accreditation</p> | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 <hr/> Page 10 of 26 |

- (a) Identity – each instance of software/hardware.
- (b) Manufacturer – includes manufacturer name, program name, and version number.
- (c) Checks - installation/operational qualifications
- (d) Location – target system name or location.
- (e) Manufacturers instructions – user manuals.
- (f) Calibrations - as discussed in 5.5.2
- (g) Maintenance Plan – N/A this is not applicable
- (h) Damage – N/A this is not applicable.

5.5 IT.6 When software test tools are used by others outside of the laboratory’s control, configurations and adaptations shall be checked and possibly reset to ensure proper functioning.

Note: For example - when another group outside of the labs’ control has access rights to the testing environment.

5.5 IT.7 Software test tools should be reset or logs emptied between test candidates to ensure that only current test data is recorded.

5.5 IT.8 Automated test cases should be checked for validity between test candidates to ensure valid test results.

5.5 IT.9 Software test tool configurations shall be safeguarded by user roles or other appropriate means.

5.6. Measurement Traceability


5.6 IT.1 Traceability is not applicable for software test tools that operate in relation to hardware processor clock cycles and /or counters with no dependence on real time.

5.7. Sampling

No additions.

5.8. Handling of test and calibration items

5.8 IT.1 Laboratories shall maintain software test candidates (SUT samples) under configuration management with appropriate metadata to ensure it is unique.

| | | |
|--|--|---|
|  World Class Accreditation | <i>The American Association for Laboratory Accreditation</i> | |
| | R214: Specific Requirements: Information Technology Testing Laboratory Accreditation Program | Document Revised: July 13, 2010 Page 11 of 26 |

5.8 IT.2 SUTs maintained under a common configuration management system accessible by customers shall be controlled and isolated.

5.9. Assuring the quality of test and calibration results

5.9 IT.1 The quality control monitoring in this clause consists of software quality control efforts documented by the lab. No other monitoring is applicable.

5.10. Reporting the results

5.10 IT.1 When test reports contain multiple tests or partial tests, the test report shall describe how they interrelate to show a complete accredited test.

5.10 IT.2 Test reports containing open errors shall have them described in an unambiguous way and should include severity descriptions in user terms.

5.10 IT.3 Any discussion of workarounds or resolution status are considered opinions and shall be denoted as such.

Document Revision History

| Date | Description |
|---------------|---|
| April 1, 2010 | Addition of requirements for use of software to support CMC claims on a scope. |
| July 13, 2010 | Changed wording in section 5.4 IT.4 (b) & (c) to represent the “current version of” ISO Guide 98 & ANSI Z540.3. |

Appendix 1

This appendix provides further description of the types of tests that fall within the Information Technology field of testing. Diagram 1 outlines the major sub disciplines within the IT field; Table 1 provides a description of each sub discipline listed in diagram 1; Table 2 provides further description and definition of the types of tests which can be performed within each sub discipline.

Diagram 1: Descriptions of the sub disciplines within the information technology field of testing

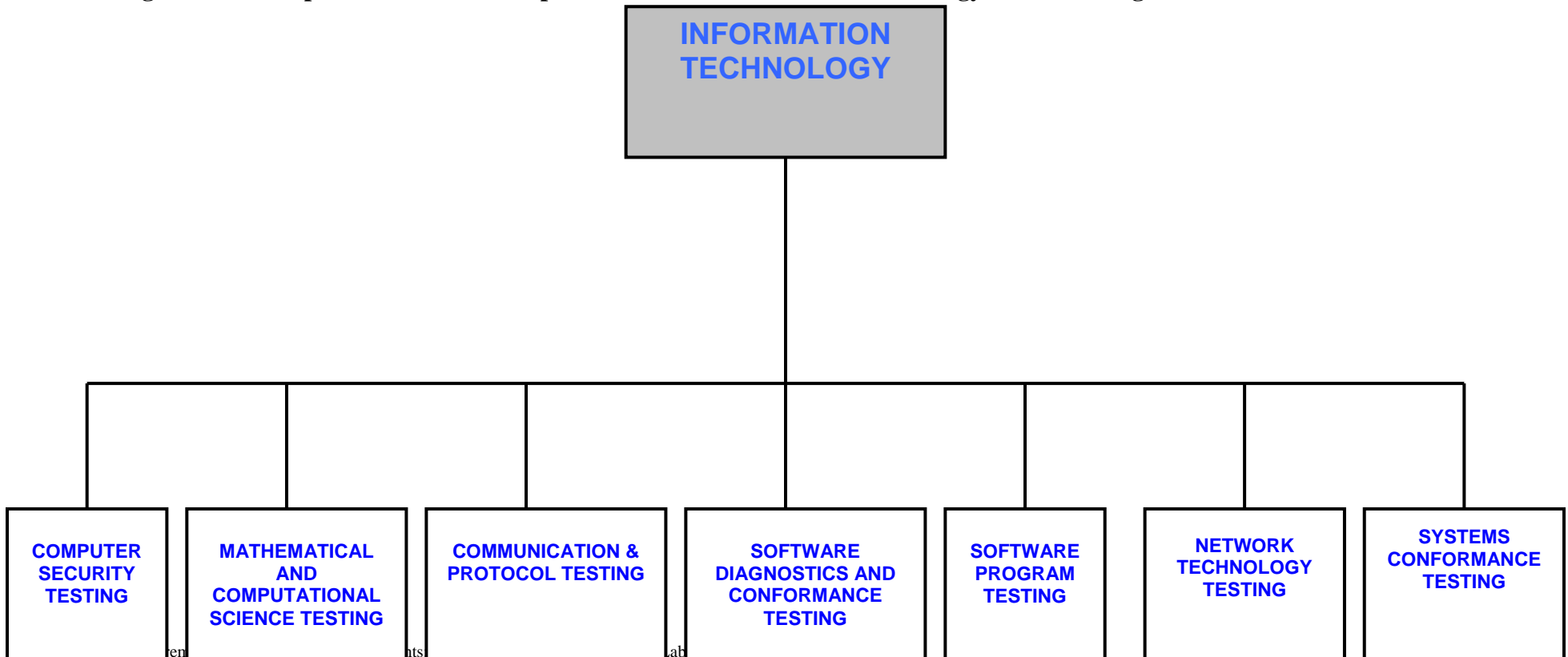


Table 1:. Sub discipline descriptions

| Sub Disciplines | Description |
|---|---|
| SOFTWARE CONFORMANCE TESTING | Testing performed to determine if a software program conforms to an external set of requirements or specifications outside of the software design. (example – regulatory or customer driven). |
| COMPUTER SECURITY TESTING | Testing performed to determine if a software application or computer system adequately conforms to specific documented security requirements or commonly recognized industry standard security requirements. It also covers MALWARE TESTING to find malicious code. |
| NETWORK TECHNOLOGY TESTING | Testing performed on network devices, components or applications to determine whether they conform to a recognized standard, customer requirements or other specifications. |
| MATHEMATICAL AND COMPUTATIONAL SCIENCE TESTING | Testing performed to evaluate data, determine probability, statistical outcomes and the precision of software algorithmic functions. |
| SOFTWARE PROGRAM TESTING | Testing as part of a software lifecycle to confirm that software meets requirements. |
| COMMUNICATION & PROTOCOL TESTING | Testing control to determine if a computer application or devices has the capability of accurately sending and receiving messages as defined in a documented specification. |
| SYSTEMS CONFORMANCE TESTING | Testing performed to determine if a system (consisting of hardware & software) conforms to an external set of requirements or specifications outside of the system design. (example – regulatory or customer driven). |

The following definitions have been compiled from the Pharmaceutical Standard Glossaries, and the Software Engineering Body of Knowledge Chapter 5, with Cross references to IEEE glossary 610.

Table 2: Test Type and Activity

| Term | Definition | IEEE 610 |
|------|--|---|
| Unit | <p>The individual testing of each module or program of an integrated application.</p> <p>This is primarily structural testing defined, planned, and conducted by (or under the direction of) IS to ascertain whether or not individual software units conform to design specifications. Often includes <i>boundary tests</i> and <i>path tests</i></p> | <p>unit. (1) A portion of a computer that constitutes the means of accomplishing some inclusive operation or function as; for example, an arithmetic unit. See also: arithmetic unit; control unit; execution unit; functional unit; logic unit; processing unit. [P10]</p> <p>unit. (2) A separately testable element specified in the design of a computer software component. [P12]</p> <p>unit. (3) A logically separable part of a computer program. [P12]</p> <p>unit. (4) A software component that is not subdivided into other components. [P12]</p> <p>unit. (5) See: test unit. [P12]</p> <p>unit. (99) Note: The terms "module," "component," and "unit" are often used interchangeably or defined to be sub-elements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized. [P12]</p> |

| Term | Definition | IEEE 610 |
|-------------------------|--|--|
| Unit (continued) | | <p>unit requirements documentation. Documentation that sets forth the functional, interface, performance, and design constraint requirements for a test unit. [P12]</p> <p>unit string. A string consisting of only one entity. [P5]</p> <p>unit testing. Testing of individual hardware or software units or groups of related units. See also: component testing; integration testing; interface testing; system testing. [P12]</p> |
| Code Coverage Analysis | The providing of a quantitative measure of test coverage. This identifies which portions of the application under test are being executed by the tests. | <p>Test Coverage of the requirements of the software item [IEEE 12207 5.3.9.3]</p> <p>test coverage. The degree to which a given test or set of tests addresses all specified requirements for a given system or component. [P12]</p> |
| Client/Server Work Flow | The functional and performance testing of the work flow in a client server system operating in its complete environment under various loads. | Nothing |
| Concurrency | The verification that an application can support multiple users accessing the same program and or data without lockouts or deadly embraces (dead-locks). | <p>Concurrent. Pertaining to the occurrence of two or more activities within the same interval of time, achieved either by interleaving the activities or by simultaneous execution. Syn: parallel (2). Contrast with: simultaneous. [P12]</p> |
| Coexistence | The verification that an application can share resources with other applications running concurrently | Nothing |

| Term | Definition | IEEE 610 |
|--|--|---|
| Code Complexity and Comprehension Analysis | The analysis of source code and indication of where the code should be revised and/or testing should focus. This counts and identifies executable paths as well as identifies dead ends and areas where there is program complexity. | <p>code review. A meeting at which software code is presented to project personnel, managers, users, customers, or other interested parties for comment or approval. Contrast with: design review; formal qualification review; requirements review; test readiness review. [P12]</p> <p>code inspection. See: inspection. [P12]</p> <p>inspection. A static analysis technique that relies on visual examination of development products to detect errors, violations of development standards, and other problems. Types include code inspection; design inspection. [P12]</p> |
| Memory | The analysis of an executing program and detection of run time memory errors including reading and using initialized memory, failing to free memory that has been allocated and over-reading or overwriting array boundaries. | <p>performance specification. A document that specifies the performance characteristics that a system or component must possess. These characteristics typically include speed, accuracy, and memory usage. Often part of a requirements specification. [P12]</p> <p>performance testing. Testing conducted to evaluate the compliance of a system or component with specified performance requirements. See also: functional testing. [P12]</p> |
| Object / Event | Testing of non-procedural language code in an event driven environment (Visual Basic, for example). | Nothing |

| Term | Definition | IEEE 610 |
|---------------|---|---|
| Functionality | The strict definition refers to the testing of the functional requirements. Inputs include normal and unexpected data (boundaries and limits). This term is also used to include security, compatibility with other external systems or networks and performance testing (see below) and usability (see below). | <p>Functional testing. (1) Testing that ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions. Syn: blackbox testing. Contrast with: structural testing. [P12]</p> <p>functional testing. (2) Testing conducted to evaluate the compliance of a system or component with specified functional requirements. See also: performance testing. [P12]</p> <p>functional unit. An entity of hardware, software, or both, capable of accomplishing a specified purpose. [P10]</p> <p>black box. (1) A system or component whose inputs, outputs, and general function are known but whose contents or implementation are unknown or irrelevant. Contrast with: glass box. [P12]</p> <p>black box. (2) Pertaining to an approach that treats a system or component as in (1). See also: encapsulation. [P12]</p> <p>black box model. A model whose inputs, outputs, and functional performance are known, but whose internal implementation is unknown or irrelevant; for example, a model of a computerized change-return mechanism in a vending machine, in the form of a table that indicates the amount of change to be returned for each amount deposited. Syn: behavioral model; input/output model. Contrast with: glass box model. [P3]</p> <p>black-box testing. See: functional testing (1). [P12]</p> |

| Term | Definition | IEEE 610 |
|-------------|--|---|
| Integration | <p>Testing conducted on a group of associated programs to ensure that inter-program communication is functioning properly. Usually run after successful Unit Testing.</p> <p>This is testing defined, planned, and conducted by (or under the direction of) IS to ascertain the degree to which the software, particularly its internal interfaces, conforms to design specifications. The testing is primarily structural from a system perspective while being functional from an individual item (unit or module or product) perspective. Previously tested items are combined (integrated) in a test environment by one of several approaches until all software has been tested together. Sample integration approaches are described below:</p> <p>big bang all items are placed in the test environment at once</p> <p>incremental items are added to the test environment one or a few at a time (usually in a “top down” or “bottom up” manner) with testing conducted in a step wise manner</p> <p>string/thread items associated with a single logical process or data flow are added to the test environment together</p> | <p>integration. The process of combining software components, hardware components, or both into an overall system. [P12]</p> <p>integration testing. Testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them. See also: component testing; interface testing; system testing; unit testing. [P12]</p> <p>component testing. Testing of individual hardware or software components or groups of related components. Syn: module testing. See also: integration testing; interface testing; system testing; unit testing. [P12]</p> <p>interface testing. Testing conducted to evaluate whether systems or components pass data and control correctly to one another. See also: component testing; integration testing; system testing; unit testing. [P12]</p> |

| Term | Definition | IEEE 610 |
|------------|--|---|
| System | <p>The testing of all aspects of a completely integrated system and its subsystems under all feasible conditions. It is the testing of all affected programs to ensure that they run correctly, that they relate to each other properly, and that the operating procedures and controls are adequate.</p> <p>This is primarily functional testing defined, planned, and conducted by (or under the direction of) IS in a production (or equivalent) environment to ascertain the degree to which the entire system (hardware and software), particularly its external interfaces, conforms to all requirements (including those for <i>security, capacity, throughput, etc.</i>) defined in the FRS.</p> | <p>system testing. Testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. See also: component testing; integration testing; interface testing; unit testing. [P12]</p> |
| Stress | <p>The use of test data chosen from the “boundaries” of input or output range classes, data structures, and procedure parameters. The choices include maximum, minimum, and trivial values or parameters.</p> | <p>stress testing. Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements. See also: boundary value. [P12]</p> |
| Structural | | <p>white-box testing. See: structural testing. [P12]</p> <p>structural testing. Testing that takes into account the internal mechanism of a system or component. Types include branch testing, path testing, statement testing. Syn: glass-box testing; white-box testing. Contrast with: functional testing (1). [P12]</p> <p>glass box. (1) A system or component whose internal contents or implementation are known. Syn: white box. Contrast with: black box. [P12]</p> <p>glass box. (2) Pertaining to an approach that treats a system or component as in (1). [P12]</p> |

| Term | Definition | IEEE 610 |
|------------------------|--|--|
| Structural (continued) | | <p>glass box model. A model whose internal implementation is known and fully visible; for example, a model of a computerized change-return mechanism in a vending machine, in the form of a diagram of the circuits and gears that make the change. Contrast with: black box model. Syn: white box model. [P3]</p> |
| Volume | <p>The running of a system with input volumes similar to those that will be encountered in actual operating conditions. This test indicates whether the system has the capacity to handle production volumes.</p> | <p>performance specification. A document that specifies the performance characteristics that a system or component must possess. These characteristics typically include speed, accuracy, and memory usage. Often part of a requirements specification. [P12]</p> <p>performance testing. Testing conducted to evaluate the compliance of a system or component with specified performance requirements. See also: functional testing. [P12]</p> |
| Performance | <p>This is similar to volume and stress testing, often combined. Performance is usually a weighted mix of throughput, response time and availability, representative models or benchmarks of average load, peak load and peak-peak load.</p> | <p>performance. The degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage. [P12]</p> <p>performance management. In networking, a management function defined for controlling and analyzing the throughput and error rate of the network. [P7]</p> |

| Term | Definition | IEEE 610 |
|-------------------------|---|---|
| Performance (continued) | | <p>performance requirement. A requirement that imposes conditions on a functional requirement; for example a requirement that specifies the speed, accuracy, or memory usage with which a given function must be performed. Contrast with: design requirement; functional requirement; implementation requirement; interface requirement; physical requirement. [P12]</p> <p>performance specification. A document that specifies the performance characteristics that a system or component must possess. These characteristics typically include speed, accuracy, and memory usage. Often part of a requirements specification. [P12]</p> <p>performance testing. Testing conducted to evaluate the compliance of a system or component with specified performance requirements. See also: functional testing. [P12]</p> |
| Usability | <p>The testing of the human factors of the system: the user's ability to interface with the system (often done using a prototype). A measure of ease of use as applied to the user interface, usually expressed in terms of the amount of training required to achieve proficiency and/or the amount of time required for a trained user to perform specific tasks.</p> | <p>user friendly. Pertaining to a computer system, device, program, or document designed with ease of use as a primary objective. Syn: user oriented. [P2]</p> <p>end user computing. The performance of system development and data processing tasks by the user of a computer system. Syn: user-driven computing. [P2]</p> <p>qualification testing. Testing conducted to determine whether a system or component is suitable for operational use. See also: acceptance testing; development testing; operational testing. [P12]</p> |

| Term | Definition | IEEE 610 |
|-----------------------|--|--|
| Usability (continued) | | <p>operational testing. Testing conducted to evaluate a system or component in its operational environment. Contrast with: development testing. See also: acceptance testing; qualification testing. [P12]</p> <p>acceptance testing. Contrast with: development testing. See also: operational testing; qualification testing. [P12]</p> <p>acceptance testing. (1) Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. [E1012]</p> <p>acceptance testing. (2) Formal testing conducted to enable a user, customer, or other authorized entity to determine whether to accept a system or component. [P12]</p> |
| Acceptance | <p>This is testing defined, planned, and conducted by the customer (user) in a production (or equivalent) environment to ascertain the degree to which the system conforms to the requirements defined in the FRS, and forms the basis for customer acceptance. Acceptance <i>test records</i> are maintained by the customer.</p> | <p>acceptance testing. Contrast with: development testing. See also: operational testing; qualification testing. [P12]</p> <p>acceptance testing. (1) Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. [E1012]</p> <p>acceptance testing. (2) Formal testing conducted to enable a user, customer, or other authorized entity to determine whether to accept a system or component. [P12]</p> |

| Term | Definition | IEEE 610 |
|--------------|---|---|
| Installation | The testing of the validity of the installation of the system (installation qualification). This includes standard set of steps to check that the installation has worked correctly. | installation and checkout phase. The period of time in the software life cycle during which a software product is integrated into its operational environment and tested in this environment to ensure that it performs as required. [P12] |
| Parallel | The running of both the old and new version of a system at the same time to identify possible differences between the two versions. | concurrent. Pertaining to the occurrence of two or more activities within the same interval of time, achieved either by interleaving the activities or by simultaneous execution. Syn: parallel (2). Contrast with: simultaneous. [P12] |
| Pilot | The testing of the software system in a full-scope, limited-size production environment. | Nothing |
| Regression | The use of pre-existing data to verify that pre-existing code is still functioning properly after a change or revision has been made. This kind of testing is used to identify unintended results of such changes to the code. This is testing defined, planned, and conducted by (or under the direction of) IS to ensure that a change results in the desired (and only the desired) effect. | regression testing. Selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements. [P12] |

| Term | Definition | IEEE 610 |
|-------------------------------|---|---|
| Supporting Definitions | | |
| Product | A logically related collection of components which represent a uniquely identified discrete element recorded in a system configuration. A product may be composed of commercial components, custom components, or both. A product may be an application program or group of application programs, an operating system, a DBMS, etc. | <p>product analysis. The process of evaluating a product by manual or automated means to determine if the product has certain characteristics. [E1002]</p> <p>product configuration identification. The current approved or conditionally approved technical documentation defining a configuration item during the production, operation, maintenance, and logistic support phases of its life cycle. It prescribes all necessary physical or form, fit, and function characteristics of a configuration item, the selected functional characteristics designated for production acceptance testing, and the production acceptance tests. Contrast with: allocated configuration identification; functional configuration identification. See also: product baseline. [P12]</p> <p>product baseline. In configuration management, the initial approved technical documentation (including, for software, the source code listing) defining a configuration item during the production, operation, maintenance, and logistic support of its life cycle. Contrast with: allocated baseline; developmental configuration; functional baseline. See also: product configuration identification. [P12]</p> |

| Term | Definition | IEEE 610 |
|------------|--|---|
| System | The hardware and software; may include several products and/or systems. | system. A collection of components organized to accomplish a specific function or set of functions. [P12] |
| Validation | <p>1: Establishing documented evidence which provides a high degree of assurance that a specific system will consistently produce a product meeting its predetermined specifications and quality attributes. (FDA)</p> <p>2: ... the process of assuring that a system or program will consistently perform as intended</p> <p>3: .. a process which documents that a computer system reproducibly performs that function it was designed to do, and which provides a high degree of assurance that the system carries out the intended functions correctly.</p> | <p>validation. The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements. Contrast with: verification. [P12]</p> <p>proof of correctness. (1) A formal technique used to prove mathematically that a computer program satisfies its specified requirements. See also: assertion; formal specification; inductive assertion method; partial correctness; total correctness. [P12]</p> <p>proof of correctness. (2) A proof that results from applying the technique in (1). [P12]</p> <p>total correctness. In proof of correctness, a designation indicating that a program's output assertions follow logically from its input assertions and processing steps, and that, in addition, the program terminates under all specified input conditions. Contrast with: partial correctness. [P12]</p> |

| Term | Definition | IEEE 610 |
|--------------|--|---|
| Verification | <p>Verification is an important activity used to establish and document the quality of work products, especially key project documents, throughout the system life cycle. Attributes such as completeness, correctness, clarity, and consistency are evaluated in assessing compliance to applicable standards in addition to fitness for use. Verification can be accomplished via either technical review.</p> | <p>verification. (1) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. Contrast with: validation. [P12]</p> <p>verification. (2) Formal proof of program correctness. See: proof of correctness. [P12]</p> <p>verification and validation. The process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements. See also: independent verification and validation. [P12]</p> |