-ilities. (1) developmental, operational, and support requirements a program must address (INCOSE Systems Engineering Handbook, 4th ed.) Note: named because they typically end in ility, such as availability, maintainability, vulnerability, reliability, supportability

1GL. (1) first-generation language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: machine language

2GL. (1) second-generation language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembly language

3D. (1) three-dimensional (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5) Syn: 3-D

3GL. (1) third-generation language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: high order language

4GL. (1) fourth-generation language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

5GL. (1) fifth-generation language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

<Viewpoint> language. (1) definitions of concepts and rules for the specification of an ODP system from the <viewpoint> viewpoint (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.2.1.1) Note: Thus, engineering language: definitions of concepts and rules for the specification of an ODP system from the engineering viewpoint.

<X> domain. (1) set of objects, each of which is related by a characterizing relationship &lt;X&gt; to a controlling object (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 10.3)

<X> federation. (1) a community of &lt;x&gt; domains (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 5.1.2)

<X> group. (1) set of objects with a particular characterizing relationship &lt;X&gt; (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 10.1)

&lt;x&gt; interceptor. (1) engineering object in a channel, placed at a boundary between &lt;x&gt; domains (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.11) Note: An &lt;x&gt; interceptor performs checks to enforce or monitor policies on permitted interactions between basic engineering objects in different domains; performs transformations to mask differences in interpretation of data by basic engineering objects in different domains. An inter-subnetwork relay is an example of an interceptor

&lt;x&gt; pattern. (1) abstract specification of a composition of objects that results in any instance of the composition having a given property, named by &lt;X&gt; (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.8)

A-0 context diagram. (1) the only context diagram that is a required for a valid IDEF0 model, the A-0 diagram contains one box, which represents the top-level function being modeled, the inputs, controls, outputs, and mechanisms attached to this box, the full model name, the model name abbreviation, the model's purpose statement, and the model's viewpoint statement (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and

A/B testing. (1) technique to determine the effectiveness of minor changes in a product or design where \&quot;A\&quot; represents the original version and \&quot;B\&quot; represents the modified version (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.1) (2)

statistical testing approach that allows testers to determine which of two systems or components performs better (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.1) Syn: split-run testing

A/IS. (1) autonomous/intelligent system (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.2)

A/IS creator. (1) person or entity that designs, develops, engineers, programs or similarly creates an autonomous/intelligent system (A/IS) (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)

ABC. (1) activity-based costing (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

abend. (1) abnormal end (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

abnormal end (abend). (1) termination of a process prior to completion (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: abort, exception

abort. (1) to terminate a process prior to completion (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: abend, exception

absent variant. (1) variant that is not determined or developed at the specific time (ISO/IEC 26554:2018 Information technology--Software and systems engineering-Tools and methods for product line testing, 3.1)

absolute address. (1) address that is permanently assigned to a device or storage location and that identifies the device or location without the need for translation or calculation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: explicit address, specific address See also: relative address, relocatable address, symbolic address, absolute assembler, absolute code, absolute instruction

absolute assembler. (1) assembler that produces absolute code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: relocating assembler

absolute code. (1) code in which all addresses are absolute addresses (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: specific code See also: relocatable code

absolute instruction. (1) computer instruction in which all addresses are absolute addresses (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: direct instruction, effective instruction, immediate instruction, indirect instruction

absolute loader. (1) loader that reads absolute machine code into main memory, beginning at the initial address assigned to the code by the assembler or compiler, and performs no address adjustments on the code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: relocating loader

abstract class. (1) class that cannot be instantiated independently (IEEE 1320.2-1998 (R2004) IEEE Standard for
Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.1) Note: That is, instantiation must
be accomplished via a subclass. A class for which every instance must also be an instance of a subclass in the cluster (a
total cluster) is called an abstract class with respect to that cluster.

abstract data type. (1) data type for which only the properties of the data and the operations to be performed on the
data are specified, without concern for how the data will be represented or how the operations will be implemented

abstract design. (1) generic form that needs specialization (further design work) to produce concrete designs
(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) design aimed at producing designs

abstraction. (1) view of an object that focuses on the information relevant to a particular purpose and ignores the
remainder of the information (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-
Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (2) process of formulating a view
(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) process of suppressing irrelevant detail to
establish a simplified model, or the result of that process (ISO/IEC 10746-2:2009 Information technology -- Open
Distributed Processing -- Reference Model: Foundations, 6.3) See also: data abstraction


acceptability. (1) exposure to loss (financial or otherwise) that an organization is willing to tolerate from a risk
(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Risk acceptability can apply to an
individual risk or to a collection of risks, such as the totality of risks confronting a project or enterprise. Acceptability can
differ for different categories of risk and can depend on the cost of treatment or other factors.

acceptability criteria. (1) documented set of characteristics of a program's work products that if satisfied, forms a
sufficient basis for judging each product's content to be acceptable to support a successful review or audit (IEEE
15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1)

acceptable. (1) meeting stakeholder expectations that can be shown to be reasonable or merited (ISO/IEC/IEEE
24765c:2014)

acceptance. (1) action by an authorized representative of the acquirer by which the acquirer assumes ownership of
products as partial or complete performance of an agreement (ISO/IEC/IEEE 24748-5:2017 Systems and software
engineering--Life cycle management--Part 5: Software development planning, 3.1)

acceptance criteria. (1) criteria that a system or component must satisfy in order to be accepted by a user,
customer, or other authorized entity (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) a set
of conditions that is required to be met before deliverables are accepted (A Guide to the Project Management Body of
Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: requirement, test criteria

acceptance test. (1) test of a system or functional unit usually performed by the purchaser on his premises after
installation with the participation of the vendor to ensure that the contractual requirements are met (ISO/IEC 2382:2015
Information technology -- Vocabulary) See also: acceptance testing, validation test

acceptance testing. (1) testing conducted to determine whether a system satisfies its acceptance criteria and to
enable the customer to determine whether to accept the system (ISO/IEC/IEEE 24765:2017 Systems and software
(2) formal testing conducted to enable a user, customer, or other authorized entity to determine whether to accept a system or component (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) See also: acceptance test, validation test

accepted deliverables. (1) products, results, or capabilities produced by a project and validated by the project customer or sponsors as meeting their specified acceptance criteria (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

access. (1) to obtain the use of a resource (ISO/IEC 2382:2015 Information technology -- Vocabulary)

access facility. (1) set of service primitives that allow a stub objects to negotiate the abstract and transfer syntax to be used for the operation data to be transmitted over the channel (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.1)

access method. (1) technique to obtain the use of data, the use of storage in order to read or write data, or the use of an input-output channel to transfer data (ISO/IEC 2382:2015 Information technology -- Vocabulary)

access routine. (1) routine that provides access to a data structure that is hidden, usually because it is a global variable or used in an abstract data type. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

access transparency. (1) distribution transparency which masks differences in data representation and invocation mechanisms to enable interworking between objects (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.4.1.1)

accessibility. (1) extent to which products, systems, services, environments and facilities can be used by people from a population with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use (ISO/IEC 25064:2013 Systems and software engineering--Software product Quality Requirements and Evaluation (SQaRE)--Common Industry Format (CIF) for usability: User needs report, 4.1) (2) degree to which a product 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 (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQaRE)--System and software quality models, 4.2.4.6) (3) usability of a product, service, environment or facility by people with the widest range of capabilities (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQaRE) -- Common Industry Format (CIF) for usability test reports, 4.1) (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.1) (4) consideration of a product, service, environment, or facility by people with the widest range of capabilities (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.2) (5) degree to which an IT service can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQaRE)--Service quality models, 3.2.2.5) (6) extent to which products, systems, services, environments and facilities can be used by people from a population with the widest range of user needs, characteristics, and capabilities to achieve identified goals in identified contexts of use (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.1) Note: Although "accessibility" typically addresses users who have disabilities, the concept is not limited to disability issues. The range of capabilities includes disabilities associated with age. Accessibility for people with...
disabilities can be specified or measured either as the extent to which a product or system can be used by users with specified disabilities to achieve specified goals with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use, or by the presence of product properties that support accessibility [ISO 25063:2014]. Context of use includes direct use or use supported by assistive technologies.

**accessibility testing.** (1) type of usability testing used to measure the degree to which a test item can be operated by users with the widest possible range of characteristics and capabilities (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.1)

**accident.** (1) unplanned event or series of events that results in death, injury, illness, environmental damage, or damage to or loss of equipment or property (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.1)

**accountability.** (1) degree to which the actions of an entity can be traced uniquely to the entity (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6.4)

**accuracy.** (1) qualitative assessment of correctness, or freedom from error (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) quantitative measure of the magnitude of error (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) in machine learning, a performance metric used to evaluate a classifier, which measures the proportion of classifications predictions that were correct (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.2) (4)

Within the quality management system, accuracy is an assessment of correctness (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition) See also: precision

**accuracy of measurement.** (1) closeness of the agreement between the result of a measurement and the true value of the measurand (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.1) Note: Accuracy is a qualitative concept. The term precision is not a synonym for "accuracy". [ISO/IEC Guide 99:2007 International vocabulary of metrology -- Basic and general concepts and associated terms] A true value is a value consistent with the definition of a given particular quantity and this is a value that would be obtained by a perfect measurement. In contexts where perfect measurement is not practically feasible, a conventional true value is a value attributed to a particular quantity and accepted, sometimes by convention, as having an uncertainty appropriate for a given purpose. 'Conventional true value', in the same reference, is sometimes called assigned value, best estimate of the value, conventional value or reference value. The accuracy can be expressed in terms of the Mean magnitude of relative error.

**ACIA.** (1) asynchronous communication interface adapter (ISO/IEC/IEEE 24765d:2015)

**ACID.** (1) Atomicity Consistency Isolation Durability (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

**acknowledged system of systems.** (1) system of systems (SoS) with recognized objectives, a designated manager, and resources for the SoS (ISO/IEC/IEEE 21841:2019 Systems and software engineering--Taxonomy of systems of systems, 3.2.1) Note: Constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches. Changes in the systems are based on cooperative agreements between the SoS and the system. Syn: acknowledged SoS
ACQ. (1) acquirer (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 4.2)

acquire resources. (1) the process of obtaining team members, facilities, equipment, materials, supplies, and other resources necessary to complete project work (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

acquirer. (1) stakeholder that acquires or procures a product or service from a supplier (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 4.1) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.1) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.1) (2) person or organization that acquires or procures a system, software product, or software service (which can be part of a system) from a supplier (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.13) (3) individual or organization that acquires or procures a system, software product or software service from a supplier (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.1) Note: The acquirer can be internal or external to the supplier organization. Acquisition of a software product can involve, but does not necessarily require, a legal contract or a financial transaction between the acquirer and supplier. Syn: buyer, owner, purchaser See also: customer


acquisition logistics. (1) technical and management activities conducted to ensure supportability implications, considered early and throughout the acquisition process to minimize support costs and to provide the user with the resources to sustain the system in the field (INCOSE Systems Engineering Handbook, 4th ed.)

acquisition strategy. (1) specific approach to acquiring products and services that is based on considerations of supply sources, acquisition methods, requirements specification types, contract or agreement types, and related acquisition risks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

action. (1) element of a step that a user performs during a procedure (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.2) (2) description of an operation to be taken in the formulation of a solution (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.7) (3) something which happens (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.3) (4) user behavior that a system accepts as a request for a particular operation (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.2)
action entry. (1) indication of the relevance of an action to a particular rule (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.9)

action of interest. (1) action in a transaction which leads to a state change of significance to the transaction (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.7.1.2)

action signature. (1) specification of an action that comprises the name for the action, the number, names and types of its parameters, and an indication of the causality of the object that instantiates the action template (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.14)

action stub. (1) list of all the actions to be taken in the solution of a problem (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.11)

activation. (1) one occurrence of a function's transformation of some subset of its inputs into some subset of its outputs (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.3)

activation constraint. (1) function's requirement for the presence of a non-empty object set in a particular arrow role as a precondition for some activation of the function (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.4)

activation function. (1) neural network formula associated with a node in a neural network that determines the output of the node (activation value) from the inputs to the neuron (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.3) Syn: transfer function


active area. (1) (on-screen documentation) area that responds to user input (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary, 4.3)

active enterprise object. (1) enterprise object that is able to fill an action role (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.1)

active interconnection. (1) physical interaction mechanism allowing the action of one thing to cause a change or to stimulate an action in another thing (ISO/IEC/IEEE 24765i:2020)

active redundancy. (1) in fault tolerance, the use of redundant elements operating simultaneously to prevent, or permit recovery from, failures (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: standby redundancy

active text. (1) text displayed on the screen that responds to user input (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

active white space. (1) area around textual or graphical elements, not including margins, which breaks up text, separates topic and subtopic groupings, indicates hierarchical and topical relationships, highlights information, or makes text easier to read (ISO/IEC/IEEE 24765a:2011)

processes, 4.1.3) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.3) (2) a distinct, scheduled portion of work performed during the course of a project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) order submitted to the system under test (SUT) by a user or an emulated user demanding the execution of a data processing operation according to a defined algorithm to produce specific output data from specific input data and (if requested) stored data (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.1) (4) defined body of work to be performed, including its required input information and output information (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (5) set of cohesive tasks of a process, which transforms inputs into outputs (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) (6) element of work performed during the implementation of a process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (7) set of actions that consume time and resources and whose performance is necessary to achieve, or contribute to, the realization of one or more outcomes (ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.1) (8) single-headed directed acyclic graph of actions, where occurrence of each action in the graph is made possible by the occurrence of all immediately preceding actions (i.e., by all adjacent actions which are closer to the head) (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.6) Note: An activity normally has an expected duration, cost, and resource requirements. Activities are often subdivided into tasks.

**activity attributes.** (1) multiple attributes associated with each schedule activity that can be included within the activity list. Activity attributes include activity codes, predecessor activities, successor activities, logical relationships, leads and lags, resource requirements, imposed dates, constraints, and assumptions. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**activity duration.** (1) the time in calendar units between the start and finish of a schedule activity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: duration

**activity group.** (1) set of related activities (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**activity list.** (1) a documented tabulation of schedule activities that shows the activity description, activity identifier, and a sufficiently detailed scope of work description so project team members understand what work is to be performed. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**activity type.** (1) classification of activities defined by the execution of the same algorithm (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.2)

**activity-based costing (ABC).** (1) cost accounting method that allocates overhead costs based on specific production activities rather than allocating from a single overhead pool (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**activity-oriented WBS.** (1) a work breakdown structure in which activities and tasks are denoted by verbs that indicate work to be accomplished. Each task name includes the work product or work products to be produced by that task. (Software Extension to the PMBOK(R) Guide Fifth Edition)

**actor.** (1) role (with respect to that action) in which the enterprise object fulfilling the role participates in the action
actual cost (AC). (1) the realized cost incurred for the work performed on an activity during a specific time period (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: actual cost of work performed (ACWP) See also: earned value management, earned value technique

actual depreciation. (1) true loss in value of an asset, determined only when the asset is sold (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It can be of interest to specify which actor initiates that action.

actual dollar analysis. (1) addressing inflation or deflation by using cash-flow amounts that represent actual amounts of money at the time of the cash flow (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: constant dollar analysis

actual duration. (1) the time in calendar units between the actual start date of the schedule activity and either the data date of the project schedule if the schedule activity is in progress or the actual finish date if the schedule activity is complete. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

actual results. (1) set of behaviors or conditions of a test item, or set of conditions of associated data or the test environment, observed as a result of test execution (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.2)

ACWP. (1) actual cost of work performed (ISO/IEC/IEEE 24765c:2014)

adaptive maintenance. (1) modification of a software product, performed after delivery, to keep a software product usable in a changed or changing environment (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.1) Note: Adaptive maintenance provides enhancements necessary to accommodate changes in the environment in which a software product must operate. These changes are those that must be made to keep pace with the changing environment.

added source statements. (1) count of source statements that were created specifically for the software product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

additive weighting. (1) assignment of different values to increase the importance of selected decision attributes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: compensatory decision technique, nondimensional scaling, analytic hierarchy process

address. (1) number, character, or group of characters that identifies a given device or storage location (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to refer to a device or storage location by an identifying number, character, or group of characters (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to deal with, to take into consideration; (specifically) to decide whether and when a defined documentation topic is to be included, either directly or by reference to another document; to decide whether an item is to be recorded prior to the test execution (in a tool or not in a tool), recorded during the test execution, recorded post-test execution, not recorded (addressed by the process), or excluded (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

address field. (1) field of a computer instruction that contains addresses, information necessary to derive addresses, or values of operands (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: address part See also: operation field

address format. (1) number and arrangement of address fields in a computer instruction. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) number and arrangement of elements within an address, such as the elements needed to identify a particular channel, device, disk sector, and record in magnetic disk storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: n-address instruction, n-plus-one address instruction

address modification. (1) arithmetic, logical, or syntactic operation performed on an address (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: effective address, indexed address, relative address, relocatable address

address space. (1) addresses that a computer program can access (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) number of memory locations that a central processing unit can address (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: In some systems, this is the set of physical storage locations that a program can access, disjoint from other programs, together with the set of virtual addresses referring to those storage locations, which are accessible by other programs.

addressing exception. (1) exception that occurs when a program calculates an address outside the bounds of the storage available to it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data
addressing mode. (1) method to search operand position in the instruction set architecture for a central processing unit (ISO/IEC/IEEE 24765:2015)

addressing range. (1) address space specified and used by the instruction system of a computer (ISO/IEC/IEEE 24765:2015) Note: An addressing range depends on the bits of address lines and addressing mode.

adjusted size. (1) a size based on the functional size multiplied by the technical complexity adjustment (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) Note: This measure does not represent functional size.

ADM. (1) architecture-driven modernization (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)

adoption process. (1) set of activities by which an organization brings CASE tools into widespread use (ISO/IEC TR 14471:2007 Information technology--Software engineering--Guidelines for the adoption of CASE tools, 2.1.2)


advanced profile. (1) profile targeted at very small enterprises (VSEs) which want to sustain and grow as an independent competitive system or software development business (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.3)

adversarial attack. (1) deliberate use of adversarial examples to cause a machine learning model to fail (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.6) Note: Typically targets machine learning models in the form of a neural network

adversarial example. (1) input to a machine learning model created by applying small perturbations to a working example that results in the model outputting an incorrect result with high confidence (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.7)

adversarial testing. (1) testing approach based on the attempted creation and execution of adversarial examples to identify defects in a machine learning model (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.8) Note: Typically applied to machine learning models in the form of a neural network


affect. (1) feelings felt by humans (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1) Note: Positive affect comprises positive feelings such as feeling calm, contented, happy, joyful, pleasant, etc. and negative affect comprises negative feelings such as feeling afraid, angry, bad, unpleasant, stressed,

afferent. (1) pertaining to a flow of data or control from a subordinate module to a superordinate module in a software system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: efferent

agent. (1) active enterprise object that has been delegated something (authorization, responsibility, provision of a
service, etc.) by, and acts for, a party (in exercising the authorization, carrying out the responsibility, providing the service, etc.) (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.8) Note: An agent can be a party or can be the ODP system or one of its components. Another system in the environment of the ODP system can also be an agent. The delegation can have been direct, by a party, or indirect, by an agent of the party having authorization from the party to so delegate.

aggregate. (1) composition of several implemented system elements that are assembled, on which a set of verification actions or validation actions is applied (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.2)

aggregate responsibility. (1) broadly stated responsibility that is eventually refined as specific properties and constraints (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.3)

aggregation. (1) derived relationship between two elements that are groups of other elements that represents all individual relationships between the grouped elements of the two groups (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)

aggregation method. (1) method that combines a set of measurement values to create a composite value (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.1) Note: Aggregation methods are based on compensatory or non-compensatory models.

agile environment. (1) organizational culture, infrastructure, and methodologies that support agile development (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 4.2)

agile team. (1) organization or team using agile development methods and approaches (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.3)

agreement. (1) mutual acknowledgment of terms and conditions under which a working relationship is conducted (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.5) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.4) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.5) (2) any document or communication that defines the initial intentions of a project. This can take the form of a contract, memorandum of understanding (MOU), letters of agreement, verbal agreements, email, etc. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: contract

AHP. (1) analytic hierarchy process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

AI. (1) artificial intelligence (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.2)

AI/S. (1) autonomous/intelligent system (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.2)

algebraic language. (1) programming language that permits the construction of statements resembling algebraic expressions, such as \( Y = X + 5 \) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: algorithmic language, list processing language, logic programming language

algorithmic language. (1) programming language designed for expressing algorithms (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: algebraic language, list processing language, logic programming language

definition. (1) alternate name for an IDEF1X model construct (class, responsibility, entity, or domain) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.4)
defined. (1) group agreement and alliance to one or more shared objectives (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: Key concepts are that each member understands critical inputs (i.e., information, context, and constraints), acts according to a plan that is communicated to all members, accepts responsibility for their part in requisite activities and tasks, and harmoniously collaborates with other members and external resources.

allocated baseline. (1) approved requirements for a product, subsystem or component, describing the functional, performance, interoperability, and interface requirements that are allocated from higher-level requirements and the verifications required to demonstrate achievement of those requirements, as established at a specific point in time and documented in the allocated configuration documentation (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1) See also: developmental configuration, functional baseline, product baseline, allocated configuration identification

allocated configuration identification. (1) in configuration management, the current approved specifications governing the development of configuration items that are part of a higher-level configuration item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Each specification defines the functional characteristics that are allocated from those of the higher-level configuration item, establishes the tests required to demonstrate achievement of its allocated functional characteristics, delineates necessary interface requirements with other associated configuration items, and establishes design constraints, if any. See also: functional configuration identification, product configuration identification, allocated baseline

allocated requirement. (1) requirement that levies all or part of the performance and functionality of a higher-level requirement on a lower level architectural element or design component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

allocation. (1) process of distributing requirements, resources, or other entities among the components of a system or program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) result of the distribution of requirements, resources, or other entities among the components of a system or program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Allocation can be made entirely to hardware, software, or humans, or to some combination to be resolved upon further functional decomposition.
allocation of an entitlement. (1) process of assigning some or all of a given entitlement to a subsidiary or other associated organizational unit which manages its own entitlement schema library (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.1) Note: The entitlement schema enables the recording of entitlement allocations.

ALM. (1) application life cycle management (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

alpha testing. (1) first stage of testing before a product is considered ready for commercial or operational use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: often performed only by users within the organization developing the software See also: beta testing

alphanumeric. (1) pertaining to data that consists of letters, digits, and usually other characters, such as punctuation marks, as well as to processes and functional units that use the data (ISO/IEC 2382:2015 Information technology -- Vocabulary)


alternate flow. (1) part of a use case that describes its alternative implementations (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It is also used to describe error conditions, since errors can be considered a kind of alternative. Syn: alternate path

alternate key. (1) candidate key of an entity other than the primary key (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.5) Note: [key style]

alternatives generation. (1) a technique used to develop as many potential options as possible in order to identify different approaches to execute and perform the work of the project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


analog. (1) pertaining to continuously variable physical quantities or to data presented in a continuous form, as well as to processes and functional units that use the data (ISO/IEC 2382:2015 Information technology -- Vocabulary)

analog computer. (1) computer whose operations are analogous to the behavior of another system and that accepts, processes, and produces analog data (ISO/IEC 2382:2015 Information technology -- Vocabulary)

analogous estimating. (1) a technique for estimating the duration or cost of an activity or a project using historical data from a similar activity or project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

analysis. (1) process of studying a system by partitioning the system into parts (functions, components, or objects) and determining how the parts relate to each other (ISO/IEC/IEEE 24765e:2015) (2) investigation and collection phase of development that aims to specify types of users and their information needs (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 4.2)

analyst. (1) member of the technical community who is skilled and trained to define problems and to analyze, develop,
and express algorithms \( (\text{ISO/IEC/IEEE 24765e:2015}) \)

 analytic hierarchy process (AHP). \( (1) \) use of matrixes to manage pair-wise relationships in decision-making (\text{ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary}) See also: additive weighting, nondimensional scaling, compensatory decision technique

 analytical techniques. \( (1) \) various techniques used to evaluate, analyze, or forecast potential outcomes based on possible variations of project or environmental variables and their relationships with other variables (\text{A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition})

 analyzability. \( (1) \) degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified (\text{ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.7.3}) \( (2) \) degree of effectiveness and efficiency with which an IT service can be analyzed for deficiencies, gaps, and failures \( (\text{ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.8.1}) \) Note: Implementation can include providing mechanisms for the product or system to analyze its own faults and provide reports before or after a failure or other event. Syn: analysability See also: modifiability

 ancestor (of a class). \( (1) \) generic ancestor of the class or a parent of the class or an ancestor of a parent of the class \( (\text{IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)}, 3.1.6) \) See also: generic ancestor, reflexive ancestor

 ancestral box. \( (1) \) box related to a specific diagram by a hierarchically consecutive sequence of one or more parent/child relationships \( (\text{IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0}, 2.1.6) \)

 ancestral diagram. \( (1) \) diagram that contains an ancestral box \( (\text{IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0}, 2.1.7) \)

 anchor point. \( (1) \) a milestone in software scheduling at which a major project life cycle transition occurs (\text{Software Extension to the PMBOK(R) Guide Fifth Edition})

 annotate. \( (1) \) command used for listing the latest version of each program's source code line, along with the date, the file version it was introduced, and the person who committed it (\text{ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary})

 annotated topic list (ATL). \( (1) \) list of all topics to be included in an information-development project with annotations that can include writer, where used, file name, and additional data (\text{ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.2})

 annotation. \( (1) \) further documentation accompanying a requirement (\text{ISO/IEC/IEEE 24765e:2015}) \( (2) \) label represented as text near to the object it is associated with (\text{ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.1})

 announcement. \( (1) \) interaction (invocation) initiated by a client object, resulting in the conveyance of information from that client object to a server object, requesting a function to be performed by that server object (\text{ISO/IEC 10746-3:2009})
annual equivalent. (1) representation of a cash flow as a series of equal annual payments (at a stated interest rate) over the planning horizon. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: AE(i) See also: future worth, present worth

annual percentage rate (APR). (1) nominal annual interest rate (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

annuity. (1) amount of a series of equal payments at regular intervals over a planning horizon (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

anomaly. (1) anything observed in the documentation or operation of a system that deviates from expectations based on previously verified system, software, or hardware products or reference documents (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)


anticipatory buffering. (1) buffering technique in which data are stored in a buffer in anticipation of a need for the data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dynamic buffering, simple buffering

anticipatory paging. (1) storage allocation technique in which pages are transferred from auxiliary storage to main storage in anticipation of a need for those pages (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: demand paging

AOA. (1) analysis of alternatives (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2) Syn: AoA

AON. (1) Activity-on-Node (ISO/IEC/IEEE 24765c:2014) See also: precedence diagramming method


aperiodic task. (1) task activated on demand (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: asynchronous task


applicability to a functional domain. (1) the ability of an FSM method to take into account the characteristics of functional user requirements (FUR) which are pertinent to FSM in a functional domain (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.2)

applicant. (1) person who has submitted an application to be admitted into the certification process (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.1)

application. (1) system for collecting, saving, processing, and presenting data by means of a computer (ISO/IEC

cohesive collection of automated procedures and data supporting a business objective, consisting of one or more components, modules, or subsystems (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.2)

Note: The term application is generally used when referring to a component of software that can be executed. It consists of one or more components, modules, or subsystems. Syn: application system, automated information system See also: application software, information system

application administration function. (1) functions performed by users which include installation, configuration, application backup, maintenance (patching and upgrading) and de-installation (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.3)

application architecture. (1) architecture including the architectural structure and rules (e.g. common rules and constraints) that constrains a specific member product within a product line (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.1)

Note: The application architecture captures the high-level design of a specific member product of a product line.

application area. (1) group of applications that have common components or characteristics, such as similar technologies or production methods, similar customers (i.e., internal versus external, government versus commercial) or a common industry sector (i.e., utilities, automotive, aerospace, information technologies) (ISO/IEC/IEEE 24765g:2018)

application asset. (1) output of a specific application engineering process (e.g. application realization) that can be exploited in other lifecycle processes of application engineering and can be adapted as a domain asset based on a product management decision (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.2)

Note: Application asset encompasses requirements, an architectural design, components, and tests.

application assets in requirements. (1) application-specific artifacts produced during application requirements engineering, such as application requirements specifications and application requirements models (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.1)

application boundary. (1) conceptual interface between the application and its users or other applications (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: The boundary defines what is external to the application; it indicates the border between the software being measured and the user; it acts as a "membrane" through which data processed by transactions pass into and out of the application. Software boundary is dependent on the user's external business view of the application. Syn: software boundary
application component. (1) component that is selected, reused or newly developed for a member product (ISO/IEC 26553:2018 Information technology-Software and systems engineering - Tools and methods for product line realization, 3.1)

application configuration. (1) derivation for member product specific executables from domain assets in realization (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.1) (2) composition results of an application by both binding variability and adding application specific variability (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.1) (3) structure of a member product, including application components and application interfaces (ISO/IEC 26555:2015 Software and systems engineering -- Tools and methods for product line technical management, 3.2)

application design. (1) process of application engineering where a single application architecture conforming to the domain architecture is derived (ISO/IEC 26550:2015 Software and systems engineering -- Reference model for product line engineering and management, 3.3)

application domain. (1) well-defined set of applications (ISO/IEC 23643:2020, Software and systems engineering -- Capabilities of software safety and security verification tools, 3.1)

application engineering. (1) life cycle consisting of a set of processes in which the application assets and member products of the product line are implemented and managed by reusing domain assets in conformance to the domain architecture and by binding the variability of the platform (ISO/IEC 26550:2015 Software and systems engineering -- Reference model for product line engineering and management, 3.4)

application engineering process. (1) processes for developing a member product in a product line (ISO/IEC 26555:2015 Software and systems engineering -- Tools and methods for product line technical management, 3.1)

application frameworks. (1) subsystem design made up of a collection of abstract and concrete classes and interfaces between them (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: Frameworks are often instantiation of a number of patterns.

application function point count. (1) a count that provides a measure of the functionality the application provides to the end-user (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) (2) activity of applying ISO/IEC 20926:2009 to measure the functional size of an application (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.4) Note: i.e., the functionality already provided to the user or that is still to be provided. With it, the effort required to support the realized application can also be determined.

application functional size. (1) measure of the functionality that an application provides to the user, determined by the application function point count (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.3) (2) size of an application expressed in function points, that is the functionality already provided to the user or that is still to be provided (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)

application generator. (1) code generator that produces programs to solve one or more problems in a particular
application area (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

application interface. (1) interface that is selected, reused or newly developed by a member product (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.3)

application management. (1) domain responsible for all of the tasks and activities that are aimed at managing, supporting, maintaining, and renewing existing applications and related data structures (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.2) Note: Application management includes all of the tasks, responsibilities, and activities that serve to bring applications into a state where they meet the requirements and needs of their owners throughout the entire life cycle of the business processes that are supported by the applications.

application management organization. (1) organizational unit that is responsible for application management for one or more applications (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.3) Note: An application management organization can be an internal or external unit in relation to the user organization.

application object. (1) component that is directly related to or forms part of an application (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.4)

application portfolio. (1) collection of applications managed by an application management organization or an entity within that application management organization (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.5) Note: The scope of the application portfolio can be the entire portfolio of that application management organization, but it can also be the applications of one or some customer organizations of entity within part of a certain customer organization.

application problem. (1) problem submitted by an end user and requiring information processing for its solution (ISO/IEC 2382:2015 Information technology -- Vocabulary)

application programming interface (API). (1) software component that allows software applications to communicate with one another (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.1)

application realization. (1) process of application engineering that develops application assets, some of which can be derived from domain assets, and member products based on the application architecture and the sets of application assets and domain assets (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.5) (2) one of the application engineering processes that includes detailed design and implementation (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.4)

application requirements analysis. (1) subprocess that understands all application specific requirements, scrutinizes incorrect and inconsistent application requirements through modelling, and then analyses and negotiates application requirements that cannot be satisfied through the domain requirements (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.3)

application requirements elicitation. (1) subprocess for identifying stakeholders relevant to an application,
eliciting application specific requirements, and binding the appropriate variants (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.2)

application requirements management. (1) subprocess that manages traceability and changes on application requirements (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.6)

application requirements specification. (1) subprocess that documents the application specific requirements and integrates it with the domain requirements specification whose variants are bound (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.4)

application requirements verification and validation. (1) subprocess that confirms that the application specific requirements are consistent and feasible and ensures that the bound variants satisfy the specific product’s requirements (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.5)

application software. (1) software designed to help users perform particular tasks or handle particular types of problems, as distinct from software that controls the computer itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary, 4.5) (2) software or a program that is specific to the solution of an application problem (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) software designed to fulfill specific needs of a user (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (4) software of an application (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.6) Note: Application software is the software that the application management organization produces, services, and maintains. There is also system software: the software to produce and maintain the application software and to run the application software on its platform. The application management organization is one of the users of the system software. See also: application software


application test asset. (1) application-specific test asset that has reuse potential (ISO/IEC 26554:2018 Information technology--Software and systems engineering-Tools and methods for product line testing, 3.3)

application testing. (1) sub-process of application engineering where domain test artefacts are reused to uncover evidence of defects in the application (ISO/IEC 26554:2018 Information technology--Software and systems engineering-Tools and methods for product line testing, 3.2)

application variability model. (1) variability model for a particular application including variability binding results, application specifically modified variability and application specifically added variability (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.2)

application-oriented language. (1) computer language with facilities or notations applicable primarily to a single application area (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: authoring language, specification language, query language

application-specific component. (1) component that is developed for a specific member product (ISO/IEC 26553:2018 Information technology-Software and systems engineering-Tools and methods for product line realization, 3.5)
application-specific requirements. (1) requirements specific to an application or requirements not covered in domain requirements (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.9) Syn: application specific requirements

applying leads and lags. (1) a technique that is used to adjust the amount of time between predecessor and successor activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

apportioned effort. (1) an activity where effort is allotted proportionately across certain discrete efforts and not divisible into discrete efforts (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

Note: Apportioned effort is one of three earned value management (EVM) types of activities used to measure work performance. See also: discrete effort

appraisal findings. (1) results of an appraisal that identify the most important issues, problems, or opportunities for process improvement within the appraisal scope (ISO/IEC/IEEE 24765:2017 Systems and software engineering -- Vocabulary) Note: Appraisal findings are inferences drawn from corroborated objective evidence.

appraisal participants. (1) members of the organizational unit who participate in providing information during an appraisal (ISO/IEC/IEEE 24765:2017 Systems and software engineering -- Vocabulary)

appraisal team leader. (1) person who leads the activities of an appraisal and has satisfied qualification criteria for experience, knowledge, and skills defined by the appraisal method (ISO/IEC/IEEE 24765:2017 Systems and software engineering -- Vocabulary)

appropriateness recognizability. (1) degree to which users can recognize whether a product or system is appropriate for their needs (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.4.1) (2) degree to which an IT service provides results that are appropriate for the user needs (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.2.1) Note: Appropriateness recognizability will depend on the ability to recognize the appropriateness of the product or system's functions from initial impressions of the product or system or any associated documentation. The information provided by the product or system can include demonstrations, tutorials, documentation or, for a web site, the information on the home page. See also: functional appropriateness

approval. (1) notification by an authorized representative that an information item appears to satisfy requirements and is complete (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.1) Note: Such approval does not shift responsibility from the supplier to meet requirements under a two-party situation.

approval authority. (1) person (or persons) or organization (or organizations) responsible for approving activities, artifacts, and other aspects of the system during its life cycle (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.5.2) (ISO/IEC/IEEE 24765h:2019) Note: The approval authority can include multiple entities, e.g. individuals or organizations. These can include different entities with different levels of approval and/or different areas of interest. In two-party situations, approval authority often rests with the acquirer. In regulatory situations, the approval authority can be a third party such as a governmental organization or its agent. In other situations, e.g. the purchase of off-the-shelf products developed by a single-party, the
independence of the approval authority can be a relevant issue to the acquirer.

**approved modification.** (1) disposition of one or more proposed changes authorizing change to any SCIs

*(ISO/IEC/IEEE 24765i:2020)* Note: There can be a many-to-many relationship of "proposed change" to "approved modification". A proposed change can cause modifications in several SCIs (even if only to the code and its test case). A modification can originate from several proposed changes, approved simultaneously or over a period of time while the modification is still in progress.

**arc.** (1) directed edge of a net which can connect a place to a transition or a transition to a place, normally represented by an arrow *(ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 2.1.1)*

**arc annotation.** (1) expression that can involve constants, variables and operators used to annotate an arc of a net *(ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 2.1.1.3)* Note: The expression evaluates to a multiset over the type of the arc's associated place.

**architect.** (1) person, team, or organization responsible for systems architecture *(ISO/IEC/IEEE 24765e:2015)*

**architecting.** (1) process of conceiving, defining, expressing, documenting, communicating, certifying proper implementation of, maintaining and improving an architecture throughout a system's life cycle *(ISO/IEC/IEEE 42010:2011 Systems and software engineering--Architecture description, 3.1)* Note: Architecting takes place in the context of an organization or a project. The entity to be architected can be of several kinds, as illustrated in the following examples: system, enterprise, solution, business, data, application, information technology, mission, product, service, software. Certifying the proper implementation of an architecture is sometimes captured as a formal statement by the architect to the client or user that the system, as built, meets the criteria as ready for use.

**architectural design.** (1) process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* result of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: functional design

**architectural design phase.** (1) life-cycle phase in which a system's general architecture is developed, thereby fulfilling the requirements laid down by the software requirements document and detailing the implementation plan in response to it *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**architectural design review.** (1) joint acquirer-supplier review to evaluate the technical adequacies of the software architectural design as depicted in the software design descriptions *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**architectural structure.** (1) physical or logical layout of the components of a system design and their internal and external connections *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**architectural style.** (1) definition of a family of systems in terms of a pattern of structural organization *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* characterization of a family of systems that are related by sharing structural and semantic properties *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*
architecture. (1) [system] fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.6) (ISO/IEC/IEEE 42010:2011 Systems and software engineering--Architecture description, 3.2) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.5) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.6) (2) set of rules to define the structure of a system and the interrelationships between its parts (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 6.6) (3) fundamental concepts or properties of an entity in its environment and governing principles for the realization and evolution of this entity and its related life cycle processes (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.3) Note: Architectures can address a wide range of concerns, expressed, for example, through architecture views and models, as illustrated in the following examples associated with particular kinds of architectures such as security architecture, functional architecture, physical architecture, resilience architecture. See also: component, module, subprogram, routine

architecture collection. (1) group of architectures held by an organization that is subject to governance and management by the organization as a whole (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.4) Note: The architectures in the collection can have relationships with each other (as in the case of product lines). The architectures in the collection can be based on the same reference architecture.

architecture description (AD). (1) work product used to express an architecture (ISO/IEC/IEEE 42010:2011 Systems and software engineering--Architecture description, 3.3) Syn: architectural description

architecture entity. (1) thing being considered, described, discussed, studied or otherwise addressed during the architecting effort (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.6) (2) thing being characterized by an architecture (ISO/IEC/IEEE 42030:2019 Software, systems, and enterprise--Architecture evaluation framework, 3.3) Note: When referring to the architecture itself of these architecture entities, it is common practice to place the name of the kind of entity in front of the word architecture. For example, the phrase system architecture is used when the thing being dealt with during the architecting effort is a system. The fundamental concepts or properties of the architecture entity are usually intended to be embodied in the entity's components, the relationships between components, and the relationships between the entity and its environment.

architecture evaluation (AE). (1) judgment about one or more architectures with respect to the specified evaluation objectives (ISO/IEC/IEEE 42030:2019 Software, systems, and enterprise--Architecture evaluation framework, 3.4)

architecture evaluation framework. (1) conventions, principles and practices for evaluating architectures (3.1) in a consistent and repeatable manner (ISO/IEC/IEEE 42030:2019 Software, systems, and enterprise--Architecture evaluation framework, 3.5) Note: This framework can be generic in nature or specific to a domain of application, a collection of concerns to be examined or a methodology. The evaluation framework can consist of different sub-architecture frameworks for an entity with many layers or levels. These could be defined and consolidated as part of the comprehensive architecture framework package. Syn: AE

architecture framework. (1) conventions, principles and practices for the description of architectures established


architecture-driven modernization (ADM). (1) process of understanding and evolving existing software assets of a system of interest (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) Note: ADM does not preclude source-to-source migrations (where appropriate), but encourages user organizations to consider modernization from an analysis and design perspective.

archival page. (1) content that is preserved as a record and not expected to change (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.1) Note: Due to technology upgrades, some archival pages cannot be readily rendered unless they are upgraded along with active pages

archive. (1) Location of system elements that are no longer present in runtime environments, but are available for examination for audit, regulatory, and other processes (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)

archiving. (1) process of placing a version of a document in a less frequently used storage area (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.3)

argument. (1) independent variable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) specific value of an independent variable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) constant, variable, or expression used in a call to a software module to specify data or program elements to be passed to...
array. (1) an n-dimensional ordered set of data items identified by a single name and one or more indices, so that each element of the set is individually addressable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

array. (1) directed line, composed of one or more connected arrow segments in a single diagram from a single source (box or diagram boundary) to a single use (box or diagram boundary) (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.8) (2) graphic presentation of a logical relationship between schedule activities in the precedence diagramming method (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: arrow segment, boundary arrow, internal arrow

array. (1) noun or noun phrase associated with an arrow segment to signify the arrow meaning of the arrow segment (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.9) Note: Specifically, an array label identifies the object type set that is represented by an arrow segment.

array. (1) object types of an object type set, regardless of how these object types can be collected, aggregated, grouped, bundled, or otherwise joined within the object type set (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.10) (2) a review conducted to evaluate the manner in which the requirements for a system have been allocated to configuration items, the system engineering process that produced the allocation, the engineering planning for the next phase of the effort, manufacturing considerations, and the planning for production engineering

array. (1) relationship between an object type set represented by an arrow segment and the activity represented by the box to which the arrow segment is attached (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.12) Note: There are four arrow roles: input, control, output, and mechanism.

array. (1) directed line that originates at a box side, arrow junction (branch or join), or diagram boundary and terminates at the next box side, arrow junction (branch or join), or diagram boundary that occurs in the path of the line (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.13)

artifact. (1) role (with respect to an action) in which the enterprise object fulfilling the role is referenced in the action (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.3) Note: An enterprise object that is an artifact in one action can be an actor in another action. Syn: artefact

artificial intelligence (AI). (1) branch of computer science devoted to developing data processing systems that perform functions normally associated with human intelligence, such as reasoning, learning, and self-improvement (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) capacity of computers or other machines to exhibit or simulate intelligent behavior (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous
and Intelligent Systems on Human Well-Being, 2.1) (3) capability of an engineered system to acquire, process and apply knowledge and skills (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.13)

artificial intelligence effect. (1) situation when a previously labelled artificial intelligence (AI) system is no longer considered to be AI as technology advances (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.10) Syn: AI effect

artificial intelligence quality metamodel. (1) metamodel intended to ensure the quality of artificial intelligence (AI)-based systems (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.11) Note: The metamodel is defined in detail in DIN SPEC 92001. Syn: AI quality metamodel

artificial intelligence-based system. (1) system including one or more components implementing artificial intelligence (AI) (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.9) Syn: AI-based system


ask. (1) combination of a specific activity; a demanded execution time, defined by a specific timeliness function; a specific task mode (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.19)

ASO. (1) Application Service Object (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

aspect. (1) special consideration within product line engineering process groups and tasks to associate specialized methods and tools (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.7) (2) special consideration within product line engineering process groups and tasks with which specialized methods and tools can be associated (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.6) (3) special consideration within product line engineering process groups and tasks to which one can associate specialized methods and tools (ISO/IEC 26554:2018 Information technology--Software and systems engineering-Tools and methods for product line testing, 3.4)

ASR. (1) alternative systems review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

assemble. (1) to translate a computer program expressed in an assembly language into its machine language equivalent (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of constructing from parts of one or more identified pieces of software (ISO/IEC/IEEE 24765j:2021) (3) activities for combining and connecting implemented system elements or aggregates to support specific goals, i.e. integration, verification, validation, manufacturing, and production (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.3) See also: compile, disassemble, interpret

assemble-and-go. (1) operating technique in which there are no stops between the assembling, linking, loading, and execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

assembled origin. (1) address of the initial storage location assigned to a computer program by an assembler, a
compiler, or a linkage editor (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) See also: loaded origin, offset (1), starting address

assembler. (1) computer program that translates programs expressed in assembly language into their machine language equivalents (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) See also: absolute assembler, compiler, cross-assembler, interpreter, relocating assembler

assembly code. (1) computer instructions and data definitions expressed in a form that can be recognized and processed by an assembler (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) Syn: assembler code See also: compiler code, interpretive code, machine code

assembly language. (1) programming language that corresponds closely to the instruction set of a given computer, allows symbolic naming of operations and addresses, and usually results in a one-to-one translation of program instructions into machine instructions (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) Syn: assembler language, low-level language, second-generation language See also: fifth-generation language, fourth-generation language, high order language, machine language.

assertion. (1) logical expression specifying a program state that must exist or a set of conditions that program variables must satisfy at a particular point during program execution. (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) (2) function or macro that complains loudly if a design assumption on which the code is based is not true (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) Note: Types include input assertion, loop assertion, output assertion. See also: invariant, proof of correctness

assessment. (1) action of applying specific documented criteria to a specific software module, package or product for the purpose of determining acceptance or release of the software module, package or product (ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools, 3.1) (2) process that evaluates a person's fulfillment of the requirements of the certification scheme (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.2)

assessment body. (1) body that performs an assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.1) Note: A body can be an organization or part of an organization that performs the assessment.

assessment constraint. (1) restriction placed on the use of the assessment outputs or on the assessment team's freedom of choice regarding the conduction of the assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.2)

assessment indicator. (1) sources of objective evidence used to support the assessors' judgment in rating process attributes (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.1)

assessment input. (1) information required before a process assessment can commence (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.3)

assessment output. (1) tangible results from an assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.4) See also: assessment record

assessment participant. (1) individual who has responsibilities within the scope of the assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.5)
assessment purpose. (1) statement, provided as part of the assessment input, which defines the reasons for performing the assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.6)

assessment record. (1) orderly, documented collection of information which is pertinent to the assessment and adds to the understanding and verification of the process profiles generated by the assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.7)

assessment scope. (1) definition of the boundaries of the assessment, provided as part of the assessment input, encompassing the boundaries of the organizational unit for the assessment, the processes to be included, the quality level for each process to be assessed, and the context within which the processes operate (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.8)

assessment sponsor. (1) individual or entity, internal or external to the organizational unit being assessed, who requires the assessment to be performed, and provides financial or other resources to carry it out (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.9)

assessment team. (1) one or more individuals who jointly perform a process assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.10)

assessor. (1) individual who participates in the rating of process attributes (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.11)

asset. (1) item, thing, or entity that has potential or actual value to an organization (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.1) (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.2) (2) item that has been designed for use in multiple contexts (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.3) (3) item, such as design, specifications, source code, documentation, test suites, or manual procedures, that has been designed for use in multiple contexts (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (4) anything that has value to a person or organization (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.1) Note: Physical assets usually refer to equipment, inventory and properties owned by the organization. Physical assets are the opposite of intangible assets, which are non-physical assets, such as leases, brands, digital assets, use rights, licenses, intellectual property rights, reputation or agreements. A grouping of assets referred to as an asset system could also be considered as an asset.

asset base. (1) reusable assets produced from both domain and application engineering (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.6)

asset life. (1) period from asset creation to asset end-of-life (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.2)

asset management. (1) coordinated activity of an organization to realize value from assets (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.3)

asset management plan. (1) documented information that specifies the activities, resources and timescales required for an individual asset, or a grouping of assets, to achieve the organization's asset management objectives
asset management system. (1) management system for asset management whose function is to establish the asset management policy and asset management objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.4)

asset portfolio. (1) assets that are within the scope of the asset management system (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.5) Note: A portfolio is typically established and assigned for managerial control purposes. Portfolios for physical assets might be defined by category (e.g. plant, equipment, tools, land). Software portfolios might be defined by software publisher, or by platform (e.g. PC, server, mainframe).

asset proposal. (1) artifact that includes major assets (functional areas and high-level common and variable features of all applications) that can be included in a product line with their quantified costs and benefits, and estimate results (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.8)

asset scoping. (1) process of identifying the potential domain assets and estimating the returns of investments in the assets (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.7) Note: Information produced during asset scoping, together with the information produced by product scoping and domain scoping, can be used to determine whether to introduce a product line into an organization.

asset system. (1) set of assets that interact or are interrelated (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.7)

asset type. (1) grouping of assets having common characteristics that distinguish those assets as a group or class (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.8)

assignment. (1) for a set of variables, the association of a value (of correct type) to each variable (ISO/IEC/IEEE 24765:2020) See also: binding

assignment statement. (1) computer program statement that assigns a value to a variable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: control statement, declaration, clear, initialize, reset

assist. (1) tester intervention in the form of direct procedural help provided by the test administrator to the test participants in order to allow the test to continue when the participants could not complete the tasks on their own (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.12)

assistive technologies. (1) hardware or software that is added to or incorporated within a system that increases accessibility for an individual (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.11)

association. (1) in UML, a relationship between an actor and a use case that indicates that the actor interacts with the system by means of the use case (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) (2)
relationship (binding) between protocol objects (or between a protocol object and an interceptor) that is established independently of the protocol exchanges that support a particular computational interaction (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.2)

**association management facility.** (1) set of service primitives which support the management of an association between protocol objects (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.3)

**associative class.** (1) class introduced to resolve a many-to-many relationship (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.7)

**associative entity.** (1) entity used to represent a relationship between other entities (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: An associative entity is used when a relationship does not otherwise provide sufficient mechanisms.

**associative entity type.** (1) entity type that contains attributes which further describe a many-to-many relationship between two other entity types (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.6) See also: entity type

**associative literal.** (1) literal that denotes an instance in terms of its value (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.8) Note: The form of expression used to state an associative literal is className with propertyName: propertyValue.

**assumption.** (1) a factor in the planning process that is considered to be true, real, or certain, without proof or demonstration (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**assumption log.** (1) a project document used to record all assumptions and constraints throughout the project life cycle (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**assurance.** (1) grounds for justified confidence that a claim has been or will be achieved (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.1.1)

**assurance case.** (1) reasoned, auditable artifact created that supports the contention that its top-level claim (or set of claims), is satisfied, including systematic argumentation and its underlying evidence and explicit assumptions that support the claim(s) (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.1.3) (2) representation of a claim or claims, and the support for these claims (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: An assurance case contains the following and their relationships: one or more claims about properties; arguments that logically link the evidence and any assumptions to the claim(s); a body of evidence and possibly assumptions supporting these arguments for the claim(s); justification of the choice of top-level claim and the method of reasoning

**assure.** (1) to promise or state with certainty by one person to another person or group (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) See also: ensure

**asynchronous.** (1) pertaining to two or more processes that do not depend upon the occurrence of specific events such as common timing (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**asynchronous communication interface adapter (ACIA).** (1) functional unit to connect interfaces for asynchronous communications (ISO/IEC/IEEE 24765d:2015)
asynchronous I/O device. (1) I/O device that generates an interrupt after producing some input or generating some output (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

asynchronous I/O device interface task. (1) task that interfaces to an I/O device and is activated by interrupts from that device (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

asynchronous message communication. (1) communication in which a producer task sends a message to a consumer task and does not wait for a response (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A message queue could build up between the tasks. Syn: loosely coupled message communication

ATL. (1) annotated topic list (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)

atomic type. (1) data type, each of whose members consists of a single, nondecomposable data item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: primitive type See also: composite type

atomicity. (1) entity at a given level of abstraction that cannot be subdivided at that level of abstraction (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 6.4)

attached process. (1) process definitions how each asset will be used in application (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.2)

attack. (1) malicious action or interaction with the system or its environment that has the potential to result in a fault or an error (and thereby possibly in a failure) or an adverse consequence (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.7)

attribute. (1) property associated with a set of real or abstract things that is some characteristic of interest (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.9) (2) inherent property or characteristic of an entity that can be distinguished quantitatively or qualitatively by human or automated means (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.1) (3) unique item of information about an entity (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) (4) identifiable association between an object and a value (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.2) (5) function from the instances of a class to the instances of the value class of the attribute (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.9) (6) label that governs the form or shape of the object it is associated with, which, in contrast to an annotation, is typically not shown as text (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.2) (7) single-valued characteristic of an entity or relationship (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: can refer either to general characteristics such as reliability, maintainability, and usability or to specific features of a software product. An attribute expresses some characteristic that is generally common to the instances of a class. The name of the attribute is the name of the role that the value class plays in describing the class, which can simply be the name of the value class (as long as using the value class name does not cause ambiguity).

attribute name. (1) role name for the value class of the attribute (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.10)
attribute sampling. (1) method of measuring quality that consists of noting the presence (or absence) of some characteristic (attribute) in each of the units under consideration (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

attributed relationship. (1) relationship that has attributes (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

attributive entity type. (1) entity type that further describes one or more attributes of another entity type (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.7) See also: entity

audience. (1) category of users sharing the same or similar characteristics and needs (for example, reason for using the documentation, tasks, education level, abilities, training, experience) (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.6) (2) category of users sharing the same or similar characteristics and needs (for example, purpose in using the documentation, tasks, education level, abilities, training, and experience) that determine the content, structure, and use of the intended documentation (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.2) Note: There can be a number of audiences for a software products documentation (for example, management, data entry, maintenance, engineering, business professionals). See also: persona

audience research. (1) planned process of interviews of representative users and analysis of interview records and personnel records (ISO/IEC/IEEE 24765a:2011) Note: The purpose of audience research is to determine the abilities, training, experience, limitations, prejudices and preferences of the intended readers of a document.

audit. (1) systematic, independent and documented process for obtaining audit evidence and evaluating it objectively to determine the extent to which audit criteria are fulfilled (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.9) (ISO/IEC 15288:2008 Systems and software engineering--System life cycle processes, 4.6) (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.7) (2) independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.10) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.9) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Lifecycle management--Part 1: Guidelines for life cycle management, 2.10) (3) systematic, independent, documented process for obtaining records, statements of fact, or other relevant information and assessing them objectively, to determine the extent to which specified requirements are fulfilled (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.7) (4) independent, continuous examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria for the purpose of providing assurance against risk (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2) Note: An audit can be an internal audit (first party) or an external audit (second party or third party), and it can be a combined or integrated audit (combining two or more disciplines). An audit results in a clear indication of whether the audit
criteria have been met. Whilst "audit" applies to management systems, "assessment" applies to conformity assessment bodies as well as more generally. Generating evidence of information technology (IT) controls that support audit is often automated where practical.

**audit team.** (1) one or more auditors conducting an audit, supported if needed by technical experts (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.10)

*Note:* One auditor of the audit team is appointed as the audit team leader. The audit team can include auditors-in-training.

**auditee.** (1) organization being audited (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.8)

**auditor.** (1) person who conducts an audit (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.9)

**augmented reality system.** (1) view of the physical world that is supplemented by computer-generated text, images, data, or other media (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.4)

**authenticity.** (1) degree to which the identity of a subject or resource can be proved to be the one claimed (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6.5)

**author check.** (1) informal review performed by the author of the work product (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.2)

**authoring environment.** (1) toolset used to create, store, and manage content units (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.5)

**authoring language.** (1) high-level programming language used to develop courseware for computer-assisted instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: authoring system

**authoring system.** (1) programming system that incorporates an authoring language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**authority.** (1) the right to apply project resources, expend funds, make decisions, or give approvals. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition)

**authorization.** (1) action indicating that a particular behavior shall not be prevented (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.4) Note: Unlike a permission, an authorization is an empowerment.

**automate.** (1) to make a process or equipment automatic (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**automated systems process.** (1) systems or software process that is performed either fully or partially supported by CASE tools (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.2.3, 2.9) Syn: assisted process, assisted software process, assisted systems process, automated process, automated software process

**automated verification system.** (1) software tool that accepts as input a computer program and a representation of its specification and produces, possibly with human help, a proof or disproof of the correctness of the program
software tool that automates part or all of the verification process.

**automatic.** (1) pertaining to a process or equipment that, under specified conditions, functions without human intervention.

**automation.** (1) conversion of processes or equipment to automatic operation, or the results of the conversion.

**autonomous growth.** (1) revealing functionality while detailing functionality that was already implied by the requirements, but was not originally recognized.

**autonomous system.** (1) system capable of working without human intervention for sustained periods.

**autonomous/intelligent system (A/IS).** (1) semi-autonomous or autonomous computer-controlled system programmed to carry out some task with or without limited human intervention, capable of decision making by independent inference and successfully adapting to its context.

**autonomy.** (1) ability of a system to work for sustained periods without human intervention.

**autonomy-based improvement.** (1) self-motivated and self-determined professional process improvement with an understanding of the work (process) objectives, latest technology, and outcomes from product use.

**availability.** (1) ability of a service or service component to perform its required function at an agreed instant or over an agreed period of time.

---

**Obtain ISO Standards:** http://www.iso.org/iso/home/store/catalogue_ics.htm

**Obtain IEEE Standards:** http://www.techstreet.com/ieee

**Obtain PMI Standards:** http://marketplace.pmi.org/Pages/default.aspx?Category=PMBOKguide

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
failure), fault tolerance and recoverability (which reflect the length of downtime following each failure). This concerns the start and finish (execution) of the application, the processing at the correct times and in the correct order, the execution of incidental processing, the opening times of online processing, and the storage period of files. See also: error tolerance, fault tolerance, reliability, robustness

BABOK. (1) Business Analysis Book of Knowledge *(ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.2)*

BAC. (1) budget at completion *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

back matter. (1) material that appears at the end of printed documentation, such as an index *(ISO/IEC/IEEE 24765a:2011)*

back-to-back testing. (1) testing in which two or more variants of a program are executed with the same inputs, the outputs are compared, and errors are analyzed in case of discrepancies *(ISO/IEC TR 19759:2016 Software Engineering - Guide to the Software Engineering Body of Knowledge (SWEBOK), 4.2.2.9) (2)* approach to testing whereby an alternative version of the system is used as a pseudo-oracle to generate expected results for comparison from the same test inputs *(ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.16) Syn: differential testing See also: mutation testing

background. (1) in job scheduling, the computing environment in which low-priority processes or those not requiring user interaction are executed *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: foreground. background processing

background processing. (1) execution of a low-priority process while higher priority processes are not using computer resources, or the execution of processes that do not require user interaction *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: foreground processing

backlog. (1) list of product requirements and deliverables not part of current work, to be prioritized and completed *(ISO/IEC/IEEE 24765h:2019) (2)* a set of software features awaiting development in a subsequent iteration *(Software Extension to the PMBOK(R) Guide Fifth Edition) (3)* collection of agile features or stories of both functional and nonfunctional requirements that are typically sorted in an order based on value priority *(ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.4)*

backout. (1) to undo the effects of a commit *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: often by introducing a new commit that restores things to their previous state

backup. (1) system, component, file, procedure, or person available to replace or help restore a primary item in the event of a failure or externally caused disaster *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2)* to create or designate a system, component, file, procedure, or person as a replacement *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: back-up

backup and recovery testing. (1) type of reliability testing that measures the degree to which system state can be restored from backup within specified parameters of time, cost, completeness, and accuracy in the event of failure *(ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.3)*

backup programmer. (1) assistant leader of a chief programmer team *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*
Responsibilities include contributing significant portions of the software being developed by the team, aiding the chief programmer in reviewing the work of other team members, substituting for the chief programmer when necessary, and having an overall technical understanding of the software being developed. See also: chief programmer

**Backus-Naur Form. (1)** formal meta-language used for defining the syntax of a language in a textual format

Backward pass. (1) a critical path method technique for calculating the late start and late finish dates by working backward through the schedule model from the project end date. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: schedule network analysis

**backward propagation. (1)** method used in artificial neural networks to determine the weights to be used on the network connections based on the computed error at the output of the network. (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.17) Note: Used to train deep neural networks

**backward recovery. (1)** reconstruction of a file to a given state by reversing all changes made to the file since it was in that state. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) type of recovery in which a system, program, database, or other system resource is restored to a previous state in which it can perform required functions. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: forward recovery

**bag. (1)** a collection class whose members are unordered but in which duplicates are meaningful (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.11) See also: list, set

**ball grid array (BGA). (1)** surface-mounted integrated circuit package with multiple connections in a grid pattern on the bottom surface. (ISO/IEC/IEEE 24765c:2014) Note: provides more connections than on packages with connectors on the edges only

**bar chart. (1)** a graphic display of schedule-related information. In the typical bar chart, schedule activities or work breakdown structure components are listed down the left side of the chart, dates are shown across the top, and activity durations are shown as date-placed horizontal bars. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: Gantt chart

**base address. (1)** address used as a reference point to which a relative address is added to determine the address of the storage location to be accessed. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: indexed address, relative address, self-relative address

**base class. (1)** relationship between a template class CB of instances of B and template class CA of instances of A, where template A is an incremental modification of template B. (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.24) See also: derived class

measurement method -- Definitions and counting guidelines for the application of function point analysis, B) (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.1) Syn: functional service, function type

**base functional component type (BFC type).** (1) defined category of Base Functional Component (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.2)

**base measure.** (1) measure defined in terms of an attribute and the method for quantifying it (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.3) Note: A base measure is functionally independent of other measures.

**base practice (BP).** (1) activity that, when consistently performed, contributes to achieving a specific process purpose (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.2)


**base value.** (1) input parameter value used in base choice testing that is normally selected based on being a representative or typical value for the parameter (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.3) Syn: base choice

**baseline.** (1) formally approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.11) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.10) (2) formally controlled and maintained set of data that serves as the basis for defining change (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1) (3) agreement or result designated and fixed at a given time, from which changes require justification and approval (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (4) snapshot of the state of a service or individual configuration items at a point in time (5) approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.11) (6) the approved version of a work product that can be changed only through formal change control procedures and is used as a basis for comparison (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (7) [verb] to establish and approve a set of data (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1) (8) gate&amp;#8208;controlled, step&amp;#8208;by&amp;#8208;step elaboration of business, budget, functional, performance, and physical characteristics, mutually agreed to by buyer and seller, and under formal change control (INCOSE Systems Engineering Handbook, 4th ed.) (9) agreed&amp;#8208;to description of the attributes of a product at a point in time, which serves as a basis for defining change (INCOSE Systems Engineering Handbook, 4th ed.) Note: Some baselines are project deliverables while others provide the basis for further work. A baseline, together with all approved changes to the baseline, represents the current approved configuration. The term is thus used to refer to a particular version of a software configuration item that has been agreed on, e.g., as a stable base for further development or to mark a specific project milestone. Baselines can be modified between formal decision gates by mutual consent
through the change control process.

**baseline data. (1)** data collected at the beginning of a process and used for comparison to subsequently collected data (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)

**baseline design. (1)** system design that has been agreed on by all stakeholders interested in the system development (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**baseline document. (1)** system or software document that defines a work product that has been placed under configuration management (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**baseline function point count. (1)** application function point count taken of the functionality at a point in time, from which changes can be measured (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

**baseline management. (1)** in configuration management, the application of technical and administrative direction to designate the documents and changes to those documents that formally identify and establish baselines at specific times during the life cycle of a configuration item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**basic engineering object. (1)** engineering object that requires the support of a distributed infrastructure (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.1) Syn: BEO

**basic flow. (1)** part of a use case that describes its most common implementation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The basic flow is written assuming that no errors or alternatives exist. Syn: basic path, happy day scenario

**basic interworking facility. (1)** a set of service primitives which have a direct correspondence with computational signals which model computational operations (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.4)

**basic maturity level. (1)** lowest level of achievement in a scale of organizational process maturity (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.3)

**basic process set. (1)** set of processes, which ensure the achievement of the basic maturity level (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.4)

**basic profile. (1)** profile targeted at VSEs developing a single application by a single work team (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.14)

**basic symbol. (1)** symbol used when the precise nature or form of, for example, the process or data media is not known or when it is not necessary to depict the actual medium (ISO 5807:1985 Information processing -- Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts, 3.1)

**basis of estimates. (1)** supporting documentation outlining the details used in establishing project estimates such as assumptions, constraints, level of detail, ranges, and confidence levels (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: basis of estimate

**basis set. (1)** set of objects used to create a multiset (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 2.1.4)

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.
batch. (1) pertaining to a system or mode of operation in which inputs are collected and processed all at one time, rather than being processed as they arrive, and a job, once started, proceeds to completion without additional input or user interaction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: conversational, interactive, online, real time

bathtub curve. (1) graph of the number of failures in a system or component as a function of time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The name is derived from the usual shape of the graph: a period of decreasing failures (the early-failure period), followed by a relatively steady period (the constant-failure period), followed by a period of increasing failures (the wearout-failure period).

Bayes’ rule. (1) statistical formula that relates the conditional probability P(A | B) to the inverse conditional probability P(B | A) (ISO/IEC/IEEE 24765j:2021)

BCWP. (1) budgeted cost of work performed (ISO/IEC/IEEE 24765c:2014)

BCWS. (1) budgeted cost of work scheduled (ISO/IEC/IEEE 24765c:2014)

behavior. (1) observable activity of a system, measurable in terms of quantifiable effects on the environment whether arising from internal or external stimulus (2) of an object, a collection of actions with a set of constraints on when they may occur (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.7) (3) peculiar reaction of a thing under given circumstances (4) aspect of an instance’s specification that is determined by the state-changing operations it can perform (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.12) Syn: behaviour

behavioral compatibility. (1) identical behavior of two objects, such that one object can replace the other with respect to a set of criteria without the environment being able to notice the difference in the objects' behavior on the basis of the set of criteria (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.4)

benchmark. (1) standard against which results can be measured or assessed (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.2) (2) procedure, problem, or test that can be used to compare systems or components to each other or to a standard (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) reference point against which comparisons can be made (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.1) (4) set of tests used to compare the performance of alternatives (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.18)

benchmark suite. (1) collection of tests used to compare the performance of alternatives (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.18)

benchmarking. (1) activity of comparing objects of interest to each other or against a benchmark to evaluate characteristic(s) (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.2) (2) the comparison of actual or planned practices, such as processes and operations, to those of comparable organizations to identify best practices, generate ideas for
improvement, and provide a basis for measuring performance (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: In the context of ISO/IEC 29155, the object of interest is IT project performance, and the characteristic is a particular aspect of an IT project such as productivity.

**benchmarking analyst.** (1) person or organization that conducts benchmarking activity (ISO/IEC 29155-3:2015 Systems and software engineering--Information technology project performance benchmarking framework--Part 3: Guidance for reporting)

**benchmarking experience base.** (1) information store that contains the evaluation of the information products and the benchmarking activity, as well as any lessons learned during benchmarking and analysis (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.3)

**benchmarking method.** (1) logical sequence of general steps to describe the process of comparing one or more attributes against a reference attribute with respect to a specified scale (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.4)

**benchmarking report.** (1) document of the results of an instance of benchmarking (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 3.7) Note: Document usually consists of various formats (e.g. textual descriptions, numeric values, statistical charts and tables), and is exchanged via various media (e.g. electronic documents, electronic data set, printed documents, and embedded data within specific computer software).

**benchmarking repository.** (1) organized and persistent data storage which is designated for benchmarking (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 3.8)

**benchmarking user.** (1) person or organization that utilizes the outcome of benchmarking (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.5)

**benefit cost analysis.** (1) in not-for-profit decision analysis, evaluating the desirability of an alternative on the ratio of the net benefits to the population to the net costs to the sponsor (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)


**beta test.** (1) second stage of testing when a product is in limited production use (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: often performed at a customer site See also: alpha testing


**BFC class.** (1) defined group of BFC types (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FiSMA 1.1 functional size measurement method, 3.1)
BFC Type. (1) a defined category of BFCs (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.2)

BGA. (1) ball grid array (ISO/IEC 24765c:2014)

bias. (1) in machine learning (ML), a measure of the distance between the predicted value provided by the ML model and a desired fair prediction (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.19)

bidder conference. (1) the meetings with prospective sellers prior to the preparation of a bid or proposal to ensure all prospective vendors have a clear and common understanding of the procurement. Also known as contractor conferences, vendor conferences, or pre-bid conferences (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

bidirectional traceability. (1) association among two or more logical entities that is discernible in either direction (to and from an entity) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: requirements traceability

big-bang testing. (1) type of integration testing in which software elements, hardware elements, or both are combined all at once into an overall system, rather than in stages (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

bill of materials (BOM). (1) documented formal hierarchical tabulation of the physical assemblies, subassemblies, and components needed to fabricate a product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

binary digit (bit). (1) unit of information that can be represented by either a zero or a one (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) element of computer storage that can hold a unit of information as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) numeral used to represent one of the two digits in the binary numeration system; zero (0) or one (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

bind. (1) to assign a value to an identifier (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

See also: dynamic binding, static binding

binder. (1) engineering object in a channel, which maintains a distributed binding between interacting basic engineering objects (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.10)

binding. (1) contractual context resulting from a given establishing behavior (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.5.2) (2) task to make a decision on relevant variants, which will be application assets, from domain assets using the domain variability model and from application assets using the application variability model (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.8) (3) task for making a decision on relevant variants using a domain variability model and decision tables (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.3) See also: assignment

binding behavior. (1) establishing behavior between two or more interfaces, and hence between their supporting

binding endpoint identifier. (1) identifier, in the naming context of a capsule, used by a basic engineering object to select one of the bindings in which it is involved, for the purpose of interaction (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.15) Note: The same form of binding endpoint identifier can be used, whether the binding involved is either local or distributed.

binding object. (1) computational object which supports a binding between a set of other computational objects (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.14)

binding precondition. (1) set of conditions required for the successful execution of a binding behavior (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.5.3)

binding time. (1) moment of variability resolution (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.3) Note: The choice of binding time is independent from variability modelling. It is the consequence of decisions made from requirements through run time. Demands for flexibility and the support of tools allow late binding times or even the use of variable binding times.

binding time decision. (1) selection for variability defined in platforms in accordance with the functional distinction between variability in time and variability in space (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.5)

binding time of variability. (1) stage when the value of variability is determined (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.7)


bit steering. (1) microprogramming technique in which the meaning of a field in a microinstruction is dependent on the value of another field in the microinstruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: immediate control See also: residual control, two-level encoding

black box. (1) system or component whose inputs, outputs, and general function are known but whose contents or implementation are unknown or irrelevant (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to an approach that treats a system or component whose inputs, outputs, and general function are known but whose contents or implementation are unknown or irrelevant (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: glass box

block. (1) group of contiguous storage locations, computer program statements, records, words, characters, or bits that are treated as a unit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to form a group (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: block-structured language, delimiter

block diagram. (1) diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) diagram of a system, computer, or device in which the principal parts are represented by suitably annotated geometrical figures to show both the functions of the parts and their functional relationships (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
configuration diagram, system resources chart (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: box diagram, bubble chart, flowchart, graph, input-process-output chart, structure chart

**block-structured language.** (1) design or programming language in which sequences of statements, called blocks, are defined, usually with begin and end delimiters, and variables or labels defined in one block are not recognized outside that block (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: structured programming language

**blocking factor.** (1) number of records, words, characters, or bits in a block (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


**BMP.** (1) bitmap image file (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 5)

**BMT.** (1) Bench Mark Test (ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools, 8.2)

**body metadata.** (1) elements in the body of an HTML document providing administrative or navigational facilities for the user or administrator (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.3)

**body of knowledge.** (1) collection of knowledge items or areas generally agreed to be essential to understanding a particular subject (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.11) Syn: BOK

**BOK.** (1) body of knowledge (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.11)

**BOM.** (1) bill of materials (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**Boolean expression.** (1) expression that evaluates to true or false (ISO/IEC/IEEE 24765i:2020)

**Boolean signature.** (1) signature where one of the sorts is Bool, corresponding to the carrier Boolean in any associated algebra, and one of the constants is true sub Bool, corresponding to the value true in the algebra (ISO/IEC/IEEE 24765i:2020)

**boot.** (1) to initialize a computer system by clearing memory and reloading the operating system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: derived from bootstrap

**boot mode.** (1) initialized mode of program operations when a computer is turned on (ISO/IEC/IEEE 24765d:2015)

**bootstrap.** (1) short computer program that is permanently resident or easily loaded into a computer and whose execution brings a larger program, such as an operating system or its loader, into memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to use a program to bring up a larger program, such as an operating system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**bootstrap loader.** (1) short computer program used to load a bootstrap (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
bottom-up. (1) pertaining to an activity that starts with the lowest-level components of a hierarchy and proceeds through progressively higher-levels (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to a method or procedure that starts at the lowest level of abstraction and proceeds towards the highest level (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: top-down. critical piece first

bottom-up design. (1) design approach in which low-level pieces of a system are combined into an overall design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of designing a system by identifying low-level components, designing each component separately, and then designing a structure to integrate the low-level components into larger and larger subsystems until the design is finished (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

bottom-up estimating. (1) a method of estimating project duration or cost by aggregating the estimates of the lower-level components of the work breakdown structure (WBS) (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

boundary. (1) conceptual interface between the software under study and its users (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.9) (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size measurement method, 3.2) (2) conceptual interface between the software being measured and its functional users (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.3) Note: The boundary provides the measurement analyst(s) with a solid delimiter to distinguish, without ambiguity, what is included inside the measured software from what is part of the measured software's operating environment. The boundary defines what is external to the application; it indicates the border between the software being measured and the user; it acts as a ?membrane? through which data processed by transactions pass into and out of the application; it is dependent on the user's external business view of the application; it is independent of non-functional and/or implementation considerations. The boundary is identical when assessing the functional size and the non-functional size. Syn: application boundary, software boundary

boundary arrow. (1) arrow with one end (source or use) not connected to any box in a diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.14) See also: internal arrow

boundary ICOM code. (1) ICOM code that maps an untunneled boundary arrow in a child diagram to an arrow attached to the parent box that is detailed by that diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.15)

boundary value. (1) data value that corresponds to a minimum or maximum input, internal, or output value specified for a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: stress testing

box. (1) rectangle containing a box name, a box number, and possibly a box detail reference and representing a function in a diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.16)

box detail reference. (1) square enclosure encompassing a box number, which indicates that the box is

**box diagram.** (1) control flow diagram consisting of a rectangle that is subdivided to show sequential steps, if-then-else conditions, repetition, and case conditions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: Chapin chart, Nassi-Shneiderman chart See also: block diagram, bubble chart, flowchart, graph, input-process-output chart, program structure diagram, structure chart

**box ICOM code.** (1) ICOM code that maps a tunneled boundary arrow to an arrow attached to some ancestral box (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.18)

**box name.** (1) verb or verb phrase placed inside a box that names the modeled function (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.19) Note: A box takes as its box name the function name of the function represented by the box. See also: function name

**box number.** (1) single digit (0, 1, 2, ..., 9) placed in the lower right corner of a box to uniquely identify that box in a diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.20) Note: The only box that can be numbered 0 is the box that represents the A0 function in A-0 and A-1 context diagrams.

**BP.** (1) base practice (ISO/IEC TR 29110-3-1:2015 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 3-1: Assessment Guide, 4.2)

**brainstorming.** (1) data-gathering and creativity technique that can be used to identify risks, ideas, or solutions to issues by using a group of team members or subject-matter experts (ISO/IEC/IEEE 24765h:2019)

**branch.** (1) computer program construct in which one of two or more alternative sets of program statements is selected for execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) point in a computer program at which one of two or more alternative sets of program statements is selected for execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) junction at which a root arrow segment (going from source to use) divides into two or more arrow segments (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.21) (4) to perform the selection in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (5) any of the alternative sets of program statements in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (6) set of evolving source file versions (ISO/IEC TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK), 6.1.3) (7) deviation from the main development line for a configuration item, which allows different persons to work on the same item at the same time (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.3) Note: Every branch is identified by a tag. Often, a branch identifies the file versions that have been or will be released as a product release. It can denote unbundling of arrow meaning, i.e., the separation of object types from an object type set. Also refers to an arrow segment into which a root arrow segment has been divided. See also: case, jump, go to, if-then-else

**branch testing.** (1) testing designed to execute each outcome of each decision point in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: path testing, statement testing

**branching.** (1) method of development in which a set of components is duplicated so the components may be
modified in parallel and optionally synchronized at a later time (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.2)

**breadcrumb trail.** (1) navigational aid with a displayed series of hyperlinks which lead from the home page to the current page, allowing the user to return to previously viewed pages (ISO/IEC/IEEE 23026:2015 Systems and software engineering -- Engineering and management of websites for systems, software, and services information, 4.4)

**break-even analysis.** (1) analysis of two or more objective functions to find where, if at all, they have the same value (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**breakpoint.** (1) point in a computer program at which execution can be suspended to permit manual or automated monitoring of program performance or results (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

*Note:* Types include code breakpoint, data breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint. A breakpoint is said to be set when both a point in the program and an event that will cause suspension of execution at that point are defined; it is said to be initiated when program execution is suspended.

**browser.** (1) application allowing a person to retrieve and read hypertext, to view the contents of hypertext nodes (Web pages), to navigate from one Web page to another, and to interact with the content, such as changing the visual appearance of the displayed content (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.5)

**BRS.** (1) business requirements specification (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)

**bubble chart.** (1) data flow, data structure, or other diagram in which entities are depicted with circles (bubbles) and relationships are represented by links drawn between the circles (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: block diagram, box diagram, flowchart, graph, input-process-output chart, structure chart

**buddy check.** (1) informal review performed independently by a colleague of the author (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.3) See also: peer review

**budget.** (1) the approved estimate for the project or any work breakdown structure component or any schedule activity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) *Note:* often used also to refer to work effort as well as, or instead of, money. See also: estimate

**budget at completion (BAC).** (1) the sum of all the budgets established for the work to be performed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**buffer.** (1) device or storage area used to store data temporarily to compensate for differences in rates of data flow, time of occurrence of events, or amounts of data that can be handled by the devices or processes involved in the transfer or use of the data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) routine that accomplishes the objectives in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to allocate, schedule, or use devices or storage areas as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**build.** (1) operational version of a system or component that incorporates a specified subset of the capabilities that the final product will provide (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software)
**burndown.** (1) an indicator of the work completed and an estimate of remaining work to be completed or remaining effort needed to complete a product development iteration cycle. Work is measured as all work done to deliver story points, stories, features, functions, function points, user stories, use cases, or requirements during a product development iteration. *(Software Extension to the PMBOK(R) Guide Fifth Edition)* See also: burnup

**burndown chart.** (1) graph that represents the work remaining to do on a project *(ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.6)*

**burndown rate.** (1) the number of software story points, features, functions, user stories, use cases, or requirements

---


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement “Copyright, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
completed per work unit (week or iteration) (Software Extension to the PMBOK(R) Guide Fifth Edition) See also: velocity

burnup. (1) an indicator of the number of story points, features, functions, user stories, use cases, or requirements
completed and the work remaining or remaining effort needed to complete a product development iteration cycle. Work is
measured as all work done to deliver story points, features, functions, user stories, use cases, or requirements during a
product development iteration. (Software Extension to the PMBOK(R) Guide Fifth Edition) Syn: burn-up See also:

burndown

bus. (1) data communication path in a computer or system (ISO/IEC/IEEE 24765c:2014)

business case. (1) a documented economic feasibility study used to establish validity of the benefits of a selected
component lacking sufficient definition and that is used as a basis for the authorization of further project management

business information management. (1) domain responsible for all of the tasks and activities that are aimed at
supporting the end users in the use of the application and at acting as the customer of the IT organizations (ISO/IEC
16350-2015 Information technology--Systems and software engineering--Application management, 4.8) Note: Note 1 to
entry: Business information management represents the business as the customer organization or client of the application
management and IT infrastructure management organizations in maintaining the functionality of the information
provisioning and the information systems. It is the demand side of the information provisioning.
Note 2 to entry: An information system may have non-automated elements such as forms and user guides. Those
elements are usually maintained by the business information management organization.

business objective. (1) strategy designed by senior management to ensure an organization’s continued existence
and enhance its profitability, market share, and other factors influencing the organization’s success (ISO/IEC TR 29110-5-
1-4:2018 Software and systems engineering-Lifecycle profiles for very small entities (VSEs)-Part 5-1-4: Software
engineering: Management and engineering guidelines: Generic profile group: Advanced profile, 3.5)

business process. (1) partially ordered set of enterprise activities that can be executed to achieve some desired
end-result in pursuit of a given objective of an organization (ISO/IEC/IEEE 12207:2017 Systems and software
engineering--Software life cycle processes, 3.1.12)

business requirements specification (BRS). (1) structured collection of the requirements (business or
mission problem or opportunity definition, concepts, and required conditions of solutions) of the business or mission and
its relation to the external environment (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle
processes-Requirements engineering, 3.1.4)

business value. (1) a documented economic feasibility study used to establish validity of the benefits of a selected
component lacking sufficient definition and that is used as a basis for the authorization of further project
strategic priorities set forth by the business as it relates to revenue, cost, risk, security, privacy, ethics, and compliance
(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build,
Package, and Deployment, 3.1)

busy. (1) pertaining to a system or component that is operational, in service, and in use (ISO/IEC/IEEE 24765:2017
Systems and software engineering-Vocabulary) See also: down, idle, up
busy time. (1) in computer performance engineering, the period of time during which a system or component is operational, in service, and in use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: operational, in service, and in use See also: down time, idle time, set-up time, up time

byte. (1) group of adjacent binary digits operated upon as a unit and usually shorter than a computer word (frequently connotes a group of eight bits) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) element of computer storage that can hold a group of bits as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) string that consists of a number of bits, treated as a unit, and usually representing a character or a part of a character (ISO/IEC 2382:2015 Information technology -- Vocabulary)

C4I. (1) command, control, communications, computer, and intelligence (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

cache. (1) temporary storage in computer memory, to improve operations by having frequently used data readily available for retrieval (ISO/IEC/IEEE 24765c:2014) (2) RAM with very high operating speed used for data storage within a processor (ISO/IEC/IEEE 24765c:2014)


cadence. (1) frequency of performing a periodic activity, such as incremental product release (Software Extension to the PMBOK(R) Guide Fifth Edition)

CAI. (1) critical application item (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

calculator. (1) device that is suitable for performing arithmetic operations, but that requires human intervention to alter its stored program, if any, and to initiate each operation or sequence of operations (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: A calculator performs some of the functions of a computer, but usually operates only with frequent human intervention.

calendar unit. (1) smallest unit of time used in scheduling a project (ISO/IEC/IEEE 24765c:2014) Note: Calendar units are generally in hours, days, or weeks, but can also be in quarter years, months, shifts, or even in minutes.

call. (1) transfer of control from one software module to another, usually with the implication that control will be returned to the calling module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) computer instruction that transfers control from one software module to another as in (1) and often specifies the parameters to be passed to and from the module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to transfer control from one software module to another as in (1) and, often, to pass parameters to the other module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) request for service(s) or action(s) with respect to an application or a related service (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.9) Note: A call might concern a request for service, information or advice; disruption or error reporting (incident); request for change; assignment (for instance an instruction to start an off-schedule production run); and complaint. See also: go to

call arrow. (1) arrow that enables the sharing of detail between IDEF0 models (linking them together) or within an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for
call by name. (1) method for passing parameters, in which the calling module provides to the called module a
symbolic expression representing the parameter to be passed, and a service routine evaluates the expression and
provides the resulting value to the called module. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary) Note: Because the expression is evaluated each time its corresponding formal parameter is used in the
called module, the value of the parameter can change during the execution of the called module. See also: call by
reference, call by value

call by reference. (1) a method for passing parameters, in which the calling module provides to the called module
the address of the parameter to be passed. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary) Note: With this method, the called module has the ability to change the value of the parameter stored by the calling
module. Syn: call by address, call by location See also: call by name, call by value

call by value. (1) method of passing parameters, in which the calling module provides to the called module the actual
value of the parameter to be passed. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note:
With this method, the called module cannot change the value of the parameter as stored by the calling module. See also:
call by name, call by reference

call graph. (1) diagram that identifies the modules in a system or computer program and shows which modules call
one another. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The result is not
necessarily the same as that shown in a structure chart. Syn: call tree, tier chart See also: structure chart, control flow
diagram, data flow diagram, data structure diagram, state diagram

call list. (1) ordered list of arguments used in a call to a software module. (ISO/IEC/IEEE 24765:2017 Systems and
software engineering-Vocabulary)

Modeling Language - Syntax and Semantics for IDEF0, 2.1.24)

called diagram. (1) decomposition diagram invoked by a calling box and identified by a page reference attached to a
IDEF0, 2.1.25)

calling box. (1) box that is detailed by a decomposition diagram that is not the box's child diagram. (IEEE 1320.1-1998
(R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.26) Note: A call arrow
is attached to the bottom of a calling box.

calling sequence. (1) sequence of computer instructions and, possibly, data necessary to perform a call to another


candidate. (1) applicant who has fulfilled prerequisites and has been admitted to the certification process. (ISO/IEC
24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1:
General requirements, 3,3)

candidate FSM method. (1) documented software size measurement method submitted for conformity evaluation
**candidate key.** (1) attribute, or combination of attributes, of an entity for which no two instances agree on the values (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.14) Note: [key style]

**capability.** (1) measure of capacity and the ability of an entity (system, person or organization) to achieve its objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems Requirements, 3.10) (2) expression of a system, product, function, or process ability to achieve a specific objective under stated conditions (INCOSE Systems Engineering Handbook, 4th ed.) (3) quality of being able to perform a given activity (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.2)

**capability maturity model.** (1) model that contains the essential elements of effective processes for one or more disciplines and describes an evolutionary improvement path from ad hoc, immature processes to disciplined, mature processes with improved quality and effectiveness (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**capable process.** (1) process that can satisfy specified product quality, service quality, and process-performance objectives (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: stable process, standard process, statistically managed process

**capacity.** (1) degree to which the maximum limits of a product or system parameter meet requirements (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.2.3) Note: Parameters can include the number of items that can be stored, the number of concurrent users, the communication bandwidth, throughput of transactions, and size of database.

**capacity testing.** (1) type of performance efficiency testing conducted to evaluate the level at which increasing load (of users, transactions, data storage, etc.) compromises a test item's ability to sustain required performance (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.5)

**capital expenditure.** (1) spending by an enterprise to acquire tangible infrastructure or facilities items, such as furniture, computers, and the like (ISO/IEC/IEEE 24765a:2011) Note: does not include acquisition of consumable supplies or of items to be included in finished products for sale

**capsule.** (1) configuration of engineering objects forming a single unit for the purpose of encapsulation of processing and storage (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.4)

**capsule manager.** (1) engineering object which manages the engineering objects in a capsule (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.5)

**CARD.** (1) cost analysis requirements description (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**cardinality.** (1) constraint on the number of entity instances that are related to the subject entity through a relationship (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (2) specification of how many
instances of a first class can or are required to exist for each instance of a second (not necessarily distinct) class, and how many instances of a second class can or are required to exist for each instance of a first class *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.15)*

**Note:** For each direction of a relationship, the cardinality can be constrained. See also: cardinality constraint

**cardinality constraint.** *(1)* constraint that limits the number of instances that can be associated with each other in a relationship *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.16)* *(2)* constraint that limits the number of members in a collection *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.16)* See also: cardinality


**case.** *(1)* single-entry, single-exit multiple-way branch that defines a control expression, specifies the processing to be performed for each value of the control expression, and returns control in all instances to the statement immediately following the overall construct *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* Computer Aided Software Engineering *(ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools, 4)* Syn: multiple exclusive selective construct See also: go to, jump, if-then-else. multiple inclusive selective construct

**CASE needs.** *(1)* organizational requirements which are met by CASE tool characteristics *(ISO/IEC TR 14471:2007 Information technology--Software engineering--Guidelines for the adoption of CASE tools, 2.1.3)* Note: These characteristics are detailed in ISO/IEC 14102:1995. They include management process, development process, maintenance, documentation, configuration management, quality assurance, verification, validation, environment needs, CASE tool integrability, quality characteristics, acquisition needs, implementation needs, support indicators, and certification requirements.

**CASE tool.** *(1)* software product that can assist software and system engineers by providing automated support for software and system engineering life-cycle activities *(ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.3)* *(2)* software product that can assist software engineers by providing automated support for software life-cycle activities *(ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools, 3.2)* Note: A CASE tool can provide support in only selected functional areas or in a wide variety of functional areas.

**cast.** *(1)* to treat an object of one type as an object of another type *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.17)* See also: coerce

**catastrophic failure.** *(1)* failure of critical software *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**categorization.** *(1)* specific way to allocate a target system into a category *(ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.5)* See also: generalization

**categorization scheme.** *(1)* orderly combination of views and categories related to software *(ISO/IEC/IEEE*
categorization space. (1) universal set of systems and software which has one or more classification axes as its individual dimension, by which stakeholder's concerns on categorization are expressed (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.6)
category. (1) specifically defined division or grouping of software based upon one or more attributes or characteristics (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) subset of categorization space, which the stakeholders are interested in, specified using a combination of one or more equivalence classes (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.9)
category entity. (1) entity whose instances represent a subtype or subclassification of another entity (generic entity) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.21) Note: [key style] See also: subclass, subtype
cause and effect diagram. (1) a decomposition technique that helps trace an undesirable effect back to its root cause (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: fishbone diagram
cautions. (1) hazardous situation which, if not avoided, can result in minor or moderate injury (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.5) (2) advisory in software user documentation that performing some action can lead to consequences that are unwanted or undefined, such as loss of data or an equipment problem (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.7) See also: danger, warning, note
CBa. (1) conduct benchmarking activity (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)
CBT. (1) computer-based training (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)
CCMS. (1) component content management system (ISO/IEC/IEEE 26531:2015 Systems and software engineering --Content management for product lifecycle, user and service management documentation, 5)


CDD. (1) capability development document (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

CDIF. (1) CASE Data Interchange Format (originally) (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 5.2)

CDIF clear text encoding. (1) clear text file encoding of a CDIF transfer file (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF exporter. (1) tool that creates a CDIF transfer file (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF family of standards. (1) set of standards that, when used together, provide a standard definition for the interchange of information between modeling tools (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF graphical notation. (1) set of rules governing the representation of CDIF modeling concepts in diagrams (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF identifier. (1) attribute that uniquely identifies an object in the model section of a transfer (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF importer. (1) tool that reads a CDIF transfer file and uses it to create or modify a model (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF meta-metamodel. (1) description of the set of concepts and notations used to define a metamodel (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Specifically, the CDIF meta-metamodel defines an Entity-Relationship-Attribute model that is used to construct and define both metamodels and the CDIF meta-metamodel itself.

CDIF metainentifier. (1) meta-meta-attribute that uniquely identifies a meta-object in the metamodel section of a transfer (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF semantic metamodel. (1) description of the set of concepts and notations used to define a model (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The CDIF semantic metamodel defines an Entity-Relationship-Attribute model that is used to construct and define models used in systems development.

CDIF transfer. (1) combination of a particular syntax, a particular encoding of that syntax, and a metamodel (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: In other words, a complete definition of the format and contents of a transfer.

CDIF transfer format. (1) combination of a particular syntax and a particular encoding of that syntax which together provides a complete definition of the transfer format (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

CDIF transfer syntax and encoding. (1) standard vehicle format supported by CDIF (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The combination of SYNTAX.1 and ENCODING.1 forms the initial CDIF transfer syntax and encoding.


CDRL. (1) contract data requirements list (ISO/IEC/IEEE 16326:2019 Systems and software engineering -- Life cycle processes -- Project management, 3)

CE. (1) Conformité' Europe' enne (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)

CEN. (1) European Committee for Standardization (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.2) Note: In French, Committee Europe' en de Normalisation

CENELEC. (1) European Committee for Electrotechnical Standardization (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.2) Note: in French, Comite' Europe'en de Normalisation E' lectrotechnique

central processing unit (CPU). (1) functional unit that consists of one or more processors and their internal storage (ISO/IEC 2382:2015 Information technology -- Vocabulary)

central tendency. (1) property of the central limit theorem predicting that the data observations in a distribution will tend to group around a central location (ISO/IEC/IEEE 24765h:2019) Note: The three typical measures of central tendency are the mean, median, and mode.

certificant. (1) recipient or holder of a certification (ISO/IEC 24773-1:2019 Software and systems engineering--Certification of software and systems engineering professionals--Part 1: General requirements, 3.4)

certificate. (1) attestation document issued by an independent third-party certification body (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.3) (2) document issued by a certification body, indicating that the named person has fulfilled the certification requirements (ISO/IEC 24773-1:2019 Software and systems engineering--Certification of software and systems engineering professionals--Part 1: General requirements, 3.5)

certification. (1) third-party attestation related to products, processes, systems, or persons (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.15) (2) formal demonstration that a system or component complies with its specified requirements and is acceptable for operational use (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) process of confirming that a system or component complies with its specified requirements and is acceptable for operational use (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: Certification is applicable to all objects of conformity
assessment except for conformity assessment bodies themselves, to which accreditation is applicable. Certification of a
management system is sometimes also called registration.

certification artifact. (1) tangible results from a certification process (ISO/IEC/IEEE 24765:2017 Systems and
software engineering-Vocabulary)

certification body. (1) third-party conformity assessment body operating certification schemes (ISO/IEC 29110-2-
1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.16)
Note: A certification body can be non-governmental or governmental (with or without regulatory authority).
certification criteria. (1) set of standards, rules, or properties to which an asset must conform in order to be certified
to a certain level (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Certification criteria
are defined by a certification policy. Certification criteria can be specified as a set of certification properties that must be met.
certification process. (1) activities by which a certification body determines that a person fulfils certification
requirements, including application, assessment, decision on certification, recertification and use of certificates and
logos/marks (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering
professionals-Part 1: General requirements, 3.6) (2) process of assessing whether an asset conforms to predetermined
certification criteria appropriate for that class of asset (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary)
certification property. (1) a statement about some feature or characteristic of an asset that can be assessed as
being true or false during a certification process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary) Note: Properties can relate to what an asset is, what it does, or how it relates to its operating environment. An
assessment of a certification quality factor is accomplished by assessing the underlying certification properties.
certification requirements. (1) set of specified requirements, including requirements of the scheme to be fulfilled
in order to establish or maintain certification of a candidate (ISO/IEC 24773-1:2019 Software and systems engineering-
Certification of software and systems engineering professionals-Part 1: General requirements, 3.7)
certification scheme. (1) competence and other requirements related to specific occupational or skilled categories
of persons (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering
professionals-Part 1: General requirements, 3.8) (2) certification system related to specified products, to which the same
specified requirements, specific rules, and procedures apply (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle
scheme addresses a candidate's knowledge, skill, competence or proficiency, but it also includes requirements for
certified person's ongoing maintenance of proficiency. A specific scheme also contains declarations concerning scope and
title; the criteria for assessment of the certified person; and declarations regarding validation of the scheme. The scheme
is documented.
certification scheme owner. (1) person or organization that is responsible for developing and maintaining a
specific certification scheme (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities
(VSEs)--Part 2-1: Framework and taxonomy, 4.18) Note: The certification scheme owner can be the certification body
itself, a governmental authority, trade association, group of certification bodies, or other.


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement
"Copyright ©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
CFD. (1) cumulative flow diagram *(Software Extension to the PMBOK(R) Guide Fifth Edition)*


chain. (1) one or more tasks submitted to the SUT in a defined sequence *(ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.3) *(2) sequence of actions within an activity where, for each adjacent pair of actions, occurrence of the first action is necessary for the occurrence of the second action *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.1)*

chain type. (1) classification of chains which is defined by the sequence of tasks types *(ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.4) Note: The emulated users submit only chains of specified chain types to the SUT.

change. (1) the modification of an existing application comprising additions, changes and deletions *(ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) *(2) add, move, modify, removal of a configuration item (CI) *(ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.4) See also: enhancement

change control. (1) a process whereby modifications to documents, deliverables, or baselines associated with the project are identified, documented, approved, or rejected *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: configuration control, version control

change control board (CCB). (1) a formally chartered group responsible for reviewing, evaluating, approving, delaying, or rejecting changes to a project, and for recording and communicating such decisions *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* Syn: change authority See also: configuration control board

change control procedure. (1) actions taken to identify, document, review, and authorize changes to a software or documentation product that is being developed *(ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.8) Note: The procedures ensure that the validity of changes is confirmed, that the effects on other items are examined, and that those people concerned with the development are notified of the changes.

change control system. (1) a set of procedures that describes how modifications to the project deliverables and documentation are managed and controlled *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

change control tools. (1) Manual or automated tools to assist with change and/or configuration management. At a minimum, the tools should support the activities of the change control board (CCB). *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

change dump. (1) selective dump of those storage locations whose contents have changed since some specified time or event *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: differential dump See also: dynamic dump, memory dump, postmortem dump, selective dump, snapshot dump, static dump
change log. (1) a comprehensive list of changes submitted during the project and their current status (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

change management. (1) judicious use of means to effect a change, or a proposed change, to a product or service (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: configuration management

change management plan. (1) A component of the project management plan that establishes the change control board, documents the extent of its authority, and describes how the change control system will be implemented. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

change package. (1) collection of objects that have been changed and approved and will be transferred to the production environment (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.10)

change project function point count. (1) a count that measures the work-output arising from modifications to an existing application that add, change or delete user functions delivered when the project is complete (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

change record. (1) record containing details of which configuration items are affected and how they are affected by an authorized change (ISO/IEC/IEEE 24765a:2011)

change request. (1) a formal proposal to modify any document, deliverable, or baseline (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) formal procedure for submitting a request for an adjustment of a configuration item (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.5) See also: modification request, request for change

change set. (1) collection of objects which can undergo change as the result of a release (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.11)

changeover system. (1) temporary information processing system used to facilitate the transition from an operational system to its successor (ISO/IEC 2382:2015 Information technology -- Vocabulary)

channel. (1) approach to distributing products and services from the original supplier to the end-user organization (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.3) (2) configuration of stubs, binders, protocol objects and interceptors providing a binding between a set of interfaces to basic engineering objects, through which interaction can occur (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.8) Note: Typical channels for software include direct, VAR, OEM, reseller, and educational reseller.

Bindings that require channels are referred to as distributed bindings in the engineering language; bindings between engineering objects that do not require channels (e.g. between engineering objects in the same cluster) are referred to as local bindings. Syn: distribution channel

channel capacity. (1) maximum amount of information that can be transferred on a given channel per unit of time; usually measured in bits per second or in baud. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: memory capacity, storage capacity

channel partner. (1) person or entity working with a software licensor or another person/entity within the channel who facilitates the sale of software to the end-user (ISO/IEC 19770-3:2016 Information technology--IT asset management--)
character. (1) letter, digit, or other symbol that is used to represent information (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) member of a set of elements that is used for the representation, organization, or control of data (ISO/IEC 2388:2015 Information technology -- Vocabulary)

collection of characters used in an encoding to represent terminal symbols (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The character set used is significant in the encoding of text and string meta-attributes for a CDIF transfer.

data type whose members can assume the values of specified characters and can be operated on by character operators, such as concatenation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: enumeration type, integer type, logical type, real type

meta-entity that provides additional attribution for another meta-object (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Other common names for characteristic entity are attributive entity and dependent entity. Each instance of a characteristic meta-entity is logically only related to one instance of one other meta-object, therefore an importer could incorporate the meta-attributes of a characteristic meta-entity with those of the 'owning' meta-object, where the owning meta-object is the one to which the characteristic meta-entity is related with a cardinality of 1:1.

a distinctive property of the FUR that is important for identifying the functional domain to which a specific set of FUR belongs (ISO/IEC TR 14143-5:2004 Information technology -- Software measurement -- Functional size measurement -- Part 5: Determination of functional domains for use with functional size measurement, 3.1)

numbering system used by a project or organization to identify costs by category, such as labor, supplies, materials, and equipment (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: code of accounts

a technique for systematically reviewing materials using a list for accuracy and completeness (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

review technique guided by a list of questions or required attributes (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.4)

testing conducted in the operational or support environment to ensure that a software product performs as required after installation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

point in a computer program at which program state, status, or results are checked or recorded (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) object template derived from the state and structure of an engineering object that can be used to instantiate another engineering object, consistent with the state of the original object at the time of checkpointing (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.20)

creating a checkpoint (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.21) Note: Checkpoints can only be created when the engineering object involved satisfies a pre-condition stated in a checkpointing policy.
chief programmer. (1) leader of a chief programmer team; a senior-level programmer whose responsibilities include producing key portions of the software assigned to the team, coordinating the activities of the team, reviewing the work of the other team members, and having an overall technical understanding of the software being developed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: backup programmer, chief programmer team

chief programmer team. (1) software development group that consists of a chief programmer, a backup programmer, a secretary/librarian, and additional programmers and specialists as needed, and that employs procedures designed to enhance group communication and to make optimum use of each member's skills (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: backup programmer, chief programmer, egoless programming


child entity. (1) entity in a specific relationship whose instances can be related to zero or one instance of the other entity (parent entity) (IEEE 1320.2-1998 R2004 IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.22) Note: [key style]
conditions (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.1.2) (3) proposition representing a requirement of the system-of-interest that enables the system-of-interest to achieve tolerable risk if it were met (ISO/IEC 15026-3:2015 Systems and software engineering -- Part 3: System integrity levels, 3.2) Note: Claims usually relate to specified versions of a product. The statement of a claim does not mean that the only possible intent or desire is to show it is true. Sometimes claims are made for the purpose of evaluating whether they are true or false or undertaking an effort to establish what is true. In its entirety, a claim is an unambiguous declaration of an assertion with any associated conditionality, giving explicit details including limitations on values and uncertainty. It could be about the future, present, or past. A safety goal is an instance of a claim.

**claims administration.** (1) the process of processing, adjudicating, and communicating contract claims (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**class.** (1) abstraction of the knowledge and behavior of a set of similar things (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.23) (2) static programming entity in an object-oriented program that contains a combination of functionality and data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) of &lt;X&gt;:s, the set of all &lt;X&gt;:s satisfying a type (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.10) Note: Classes are used to represent the notion of "things whose knowledge or actions are relevant." See also: type

**class hierarchy.** (1) ordering of classes, in which a subclass is a specialization of its superclass (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A class inherits attributes and relationships from its superclass and can define additional attributes and relationships of its own.

**class-level attribute.** (1) mapping from the class itself to the instances of a value class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.24)

**class-level operation.** (1) mapping from the (cross product of the) class itself and the instances of the input argument types to the (cross product of the) instances of the other (output) argument types (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.25)

**class-level responsibility.** (1) responsibility that represents some aspect of the knowledge, behavior, or rules of the class as a whole (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.26) See also: instance-level responsibility

**classification.** (1) machine learning function that predicts the output class for a given input (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.20)

**classification axis.** (1) total range of a mapping of systems and software for categorizing them from a particular perspective (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.7)

**classifier.** (1) machine learning model used for classification (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.20)

**clear.** (1) to set a variable, register, or other storage location to zero, blank, or other null value (ISO/IEC/IEEE
clear text file encoding. (1) class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

clear text file encoding. (1) class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

client. (1) code or process that invokes an operation on an object (ISO/IEC 19500-2:2012 Information technology -- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.3) (2) of a service, any entity capable of requesting the service (ISO/IEC 19500-1:2012 Information technology--Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3) (3) for certification, the organization that is responsible to a certification body for ensuring certification requirements, including product requirements, are fulfilled (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.19)
class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

client object. (1) object which requests that a service be performed by another object (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.4.5)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

client-side. (1) node, cluster or capsule, which: a) contains a basic engineering object corresponding to a computational client object; and b) contains, or is potentially capable of containing, stub, binder and protocol objects in a channel supporting operations involving the client object (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.5)

client-side. (1) node, cluster or capsule, which: a) contains a basic engineering object corresponding to a computational client object; and b) contains, or is potentially capable of containing, stub, binder and protocol objects in a channel supporting operations involving the client object (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.5)

clock pulse generator (CPG). (1) electronic unit to produce uniform timed signals (ISO/IEC/IEEE 24765d:2015)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

clock pulse generator (CPG). (1) electronic unit to produce uniform timed signals (ISO/IEC/IEEE 24765d:2015)

cloning. (1) instantiating a cluster from a cluster checkpoint (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.24)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

Close Procurements. (1) the process of completing each project procurement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

Close Procurements. (1) the process of completing each project procurement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

Close Project or Phase. (1) the process of finalizing all activities for the project, phase, or contract (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

closed loop. (1) loop that has no exit and whose execution can be interrupted only by intervention from outside the computer program or procedure in which the loop is located (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: UNTIL, WHILE

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

closed subroutine. (1) subroutine that is stored at one given location rather than being copied into a computer program at each place that it is called (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: open subroutine

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

closed subroutine. (1) subroutine that is stored at one given location rather than being copied into a computer program at each place that it is called (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: open subroutine

closed term. (1) term comprising constants and operators but no variables (ISO/IEC/IEEE 24765i:2020) Syn: ground term

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

class of techniques for representing data based on first defining a human readable representation using some specific character repertoire and then defining an encoding for that repertoire (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

cluster. (1) configuration of basic engineering objects forming a single unit for the purposes of deactivation, checkpointing, reactivation, recovery and migration (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.2)

closed subroutine. (1) subroutine that is stored at one given location rather than being copied into a computer program at each place that it is called (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: open subroutine

cluster checkpoint. (1) cluster template containing checkpoints of the basic engineering objects in a cluster (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.22)
cluster manager. (1) engineering object which manages the basic engineering objects in a cluster (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.3)

cluster template. (1) object template for a configuration of objects and any activity required to instantiate those objects and establish initial bindings (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.19)

clustering. (1) grouping of a set of objects such that objects in the same group (i.e. a cluster) are more similar to each other than to those in other clusters (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.22)


CM tool. (1) software product that can assist software engineers by providing automated support for configuration management activities (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.9) Syn: configuration management tool

CMDB. (1) configuration management database (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2)


CMP. (1) configuration management plan (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2)


CO. (1) service control process (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 4.1)

co-existence. (1) degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality requirements and evaluation, 6.2.2)
software quality models, 4.2.3.1) Syn: coexistence

code. (1) in software engineering, computer instructions and data definitions expressed in a programming language or in a form output by an assembler, compiler, or other translator (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to express a computer program in a programming language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) character or bit pattern that is assigned a particular meaning (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: source code, object code, machine code, micro code

code breakpoint. (1) breakpoint that is initiated upon execution of a given computer instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: control breakpoint See also: data breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint

code data. (1) type of data entities used for software sizing (in addition to business data and reference data) (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: Code data usually exists to satisfy non-functional requirements (NFR) from the user (for quality requirements, physical implementation or a technical reason). Code data provides a list of valid values that a descriptive attribute may have. Typically, the attributes of the code data are Code, Description or other standard attributes describing the code; e.g., standard abbreviation, effective date, termination date, audit trail data.

code freeze. (1) period during which non-critical changes to the code are not allowed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

code generator. (1) software tool that accepts as input the requirements or design for a computer program and produces source code that implements the requirements or design generator (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: application
code of accounts. (1) a numbering system used to uniquely identify each component of the work breakdown structure (WBS) (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: chart of accounts

code of ethics standard. (1) standard that describes the characteristics of a set of moral principles dealing with accepted standards of conduct by, within, and among professionals (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

code review. (1) meeting at which software code is presented to project personnel, managers, users, customers, or other interested parties for comment or approval (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: design review, formal qualification review, requirements review, test readiness review

code tuning. (1) process of making statement-level changes to a program to make it more efficient (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) making changes to program source code to optimize performance, usually to increase speed or reduce memory usage (3) changes made to program source code for the purpose of optimizing performance, usually to increase speed or reduce memory usage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

coding. (1) in software engineering, the process of expressing a computer program in a programming language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) transforming of logic and data from
design specifications (design descriptions) into a programming language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

do coerc. (1) to treat an object of one type as an object of another type by using a different object (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.28) See also: cast

cohesion. (1) manner and degree to which the tasks performed by a single software module are related to one another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in software design, a measure of the strength of association of the elements within a module (ISO/IEC TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK)) Note: Types include coincidental, communicational, functional, logical, procedural, sequential, and temporal. Syn: module strength See also: coupling coincidental cohesion. (1) type of cohesion in which the tasks performed by a software module have no functional relationship to one another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: communicational cohesion, functional cohesion, logical cohesion, procedural cohesion, sequential cohesion, temporal cohesion

collaboration. (1) cooperative exchange of requests among classes and instances in order to achieve some goal (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.29)
collaborative system of systems. (1) system of systems (SoS) in which component systems interact more or less voluntarily to fulfill agreed-upon central purposes (ISO/IEC/IEEE 21841:2019 Systems and software engineering--Taxonomy of systems of systems, 3.2.2) Note: Constituent systems collectively decide how to provide or deny service, thereby providing means of enforcing and maintaining consistency. Syn: collaborative SoS

collapse. (1) to terminate development on one branch by integrating it with another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

Collect Requirements. (1) the process of determining, documenting, and managing stakeholder needs and requirements to meet project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
collection cardinality. (1) specification, for a collection-valued property, of how many members the value of the property, that is, the collection, can or is required to have for each instance (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.30) See also: cardinality constraint

collection class. (1) class in which each instance is a group of instances of other classes (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.31)
collection-valued. (1) value that is complex (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.33) Note: That is, having constituent parts. See also: scalar

collection-valued class. (1) class in which each instance is a collection of values (IEEE 1320.2-1998 (R2004) IEEE
collection-valued property. (1) property that maps to a collection class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.34) See also: scalar-valued class

colocational. (1) an organizational placement strategy where the project team members are physically located close to one another in order to improve communication, working relationships, and productivity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: co-location

combinatorial testing. (1) closed-box test design technique in which test cases are designed to execute specific combinations of values of several parameters (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.23)

comfort. (1) degree to which the user is satisfied with physical comfort (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.3.4)

command. (1) expression that can be input to a computer system to initiate an action or affect the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

command language. (1) language used to express commands to a computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: command-driven

command-driven. (1) pertaining to a system or mode of operation in which the user directs the system through commands (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: command driven See also: menu-driven

comment. (1) information embedded within a computer program, job control statements, or a set of data that provides clarification to human readers but does not affect machine interpretation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

commercial-off-the-shelf (COTS). (1) [software] product available for purchase and use without the need to conduct development activities (ISO/IEC/IEEE 90003:2018 Software engineering -- Guidelines for the application of ISO 9001:2015 to computer software, 3.4) (2) commercial item that requires no unique acquirer modifications or maintenance over the life cycle of the product to meet the needs of the procuring agency (INCOSE Systems Engineering Handbook, 4th ed.) Note: COTS software product includes the product description (including all cover information, data sheet, web site information, etc.), the user documentation (necessary to install and use the software), the software contained on a computer sensible media (disk, CD-ROM, internet downloadable, etc.). Software is mainly composed of programs and data. This definition applies also to product descriptions, user documentation and software which are produced and supported as separate manufactured goods, but for which typical commercial fees and licensing considerations do not apply. Syn: commercial off the shelf, Commercial-Off-The-Shelf See also: software product

commercial-off-the-shelf software product. (1) software product defined by a market-driven need, commercially available, and whose fitness for use has been demonstrated by a broad spectrum of commercial users (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation)
(SQuaRE)–Evaluation process, 4.6) Syn: COTS software product

commissioning. (1) procedures prior, or related, to the handing over of a product ready to be placed into service (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.2) Note: Commissioning may include final acceptance testing, the handing over of relevant documentation for the supported product, or instructing personnel.

commit. (1) to integrate the changes made to a developer's private view of the source code into a branch accessible through the version control system's repository (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

commit message. (1) explanatory message accompanying a commit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: often contains a brief description of the change and its rationale; names of contributors, reviewers, or approvers; a reference to third-party software from which the change was obtained; a schedule for integrating it to other branches; and a reference to the issue identifier associated with the change

commit privileges. (1) person's authority to commit changes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Sometimes privileges are associated with a specific part of the product (for example, artwork or documentation) or a specific branch.

commit war. (1) series of conflicting and mutually reversing commits introduced by developers who disagree on how a particular element is being coded (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: sometimes starts with a hostile backout.

commit window. (1) period during which commits are allowed for a specific branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: In some development environments, commit windows for a maintenance branch might only open for short periods a few times a year.

commitment. (1) action resulting in an obligation by one or more of the participants in the act to comply with a rule or perform a contract (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.2) Note: The enterprise object(s) participating in an action of commitment can be parties or agents acting on behalf of a party or parties. In the case of an action of commitment by an agent, the principal becomes obligated.

committer. (1) developer with commit privileges (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

common ancestor constraint. (1) constraint that involves two or more relationship paths to the same ancestor class and states either that a descendent instance must be related to the same ancestor instance through each path or that it must be related to a different ancestor instance through each path (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.36)

common cause. (1) source of variation of a process that exists because of normal and expected interactions among components of a process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: On a control chart, it appears as part of the random process variation (i.e., variation from a process that would be considered normal or not unusual), and is indicated by a random pattern of points within the control limits. Syn: random cause See also: special cause
common storage. (1) portion of main storage that can be accessed by two or more modules in a software system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: common area, common block See also: global data

common-environment coupling. (1) type of coupling in which two software modules access a common data area (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: common coupling, common environment coupling See also: content coupling, control coupling, data coupling, hybrid coupling, pathological coupling

commonality. (1) set of functional and non-functional characteristics that is shared by all applications belonging to the product line (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.9) (2) functional and nonfunctional characteristics that can be shared with all member products within a product line (INCOSE Systems Engineering Handbook, 4th ed.)

communication constraints. (1) restrictions on the content, timing, audience, or individual who will deliver a communication, usually stemming from specific legislation or regulation, technology, or organizational policies (ISO/IEC/IEEE 24765h:2019)

communication interface. (1) interface of a protocol object that can be bound to an interface of either an interceptor object or another protocol object at an interworking reference point (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.14)

communication management. (1) management of objects which support the communication between objects within an ODP system (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 14.2)

communication management plan. (1) a component of the project, program, or portfolio management plan that describes how, when, and by whom information about the project will be administered and disseminated (A Guide to the Project Management Body of Knowledge (PMBOK) Guide -- Sixth Edition) Syn: communications management plan

communication methods. (1) a systematic procedure, technique, or process used to transfer information among project stakeholders (A Guide to the Project Management Body of Knowledge (PMBOK) Guide -- Sixth Edition)

communication models. (1) a description, analogy or schematic used to represent how the communication process will be performed for the project (A Guide to the Project Management Body of Knowledge (PMBOK) Guide -- Sixth Edition) Syn: communication model

communication requirements analysis. (1) an analytical technique to determine the information needs of the project stakeholders through interviews, workshops, study of lessons learned from previous projects, etc. (A Guide to the Project Management Body of Knowledge (PMBOK) Guide -- Sixth Edition)

communication styles assessment. (1) a technique to identify the preferred communication method, format, and content for stakeholders for planned communication activities (A Guide to the Project Management Body of Knowledge (PMBOK) Guide -- Sixth Edition)

communication technology (CT). (1) specific tools, systems, computer programs, etc., used to transfer information among project stakeholders (A Guide to the Project Management Body of Knowledge (PMBOK) Guide -- Sixth Edition)

communicational cohesion. (1) type of cohesion in which the tasks performed by a software module use the...
same input data or contribute to producing the same output data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: coincidental cohesion, functional cohesion, logical cohesion, procedural cohesion, sequential cohesion, temporal cohesion communications domain. (1) set of protocol objects capable of interworking (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.13) communications planning. (1) process of defining how to meet the information and communication needs of the stakeholders: who needs what information, when they need it, and how it will be given to them (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) community. (1) configuration of objects formed to meet an objective (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 5.1.1) Note: The objective is expressed as a contract which specifies how the objective can be met. community object. (1) composite enterprise object that represents a community (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.2.2) Note: Components of a community object are objects of the community represented compact disc read only memory (CD-ROM). (1) optical disk which can be read, but not erased or rewritten (ISO/IEC/IEEE 24765c:2014) compaction. (1) in microprogramming, the process of converting a microprogram into a functionally equivalent microprogram that is faster or shorter than the original (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: local compaction, global compaction comparator. (1) software tool that compares two computer programs, files, or sets of data to identify commonalities or differences (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Typical objects of comparison are similar versions of source code, object code, database files, or test results. compatibility. (1) degree to which a product, system or component can exchange information with other products, systems or components, or perform its required functions, while sharing the same hardware or software environment (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.3) (2) ability of two or more systems or components to exchange information (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) capability of a functional unit to meet the requirements of a specified interface without appreciable modification (ISO/IEC 2382:2015 Information technology -- Vocabulary) compatibility testing. (1) type of testing that measures the degree to which a test item can function satisfactorily alongside other independent products in a shared environment (co-existence), and where necessary, exchanges information with other systems or components (interoperability) (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.6) compensatory decision technique. (1) multiple-attribute decision technique that allows better performance in some of the attributes to compensate for lower performance in one or more of the other attributes; use of trade-offs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: noncompensatory decision technique, additive weighting, analytic hierarchy process, nondimensional scaling
compensatory model. (1) multiple-criteria decision-making model, in which a composite measure is composed of individually weighted terms and where criteria (also referred to as attribute terms) with a high value can compensate for those of a low value in proportion to each weight (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.2) Note: A compensatory model suggests that improving the more important measures (those with a higher weighting) is more likely to increase or improve the overall composite value than improving the less important ones. This model assumes that the weight (influence level) of criteria remains the same regardless of the measured level of the criteria.

competence. (1) ability to apply knowledge and skills to achieve intended results (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.11) (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.9) (2) ability to demonstrate and apply the combination of knowledge, formal and informal skills, training, experience, and behavioral attributes to achieve intended organizational and technical results (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: Results are defined with respect to tasks, functions or responsibilities, which in turn are job/role/title-related. Competence can be used to refer to general ability (e.g. overall competence), while competency can be used to refer to a specific ability (e.g. competency in design of user interfaces). Syn: competency See also: competent

competent. (1) having the combination of knowledge, formal and informal skills, training, experience, and behavioral attributes required to perform a task or role (ISO/IEC/IEEE 24765c:2014) See also: competent


competent person. (1) person who has acquired through training, qualifications or experience, or a combination of these, the knowledge and skills enabling that person to perform a specified task (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.3) See also: skilled person

compile. (1) to translate a computer program expressed in a high-order language into its machine language equivalent (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assemble, decompile, interpret

compile-and-go. (1) operating technique in which there are no stops between the compiling, linking, loading, and execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

compiler. (1) computer program that translates programs expressed in a high-order language into their machine language equivalents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembler, interpreter, cross-compiler, incremental compiler, root compiler

compiler code. (1) computer instructions and data definitions expressed in a form that can be recognized and processed by a compiler (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembly code, interpretive code, machine code

compiler directive source statement. (1) source statement that defines macros, or labels, or directs the
compiler to insert external source statements (for example, an include statement), or directs conditional compilation, or is not described by one of the other type attributes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**compiler generator.** (1) translator or interpreter used to construct part or all of a compiler (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: compiler, metacompiler

**complaint.** (1) record of perceived non-compliance with a service level agreement or customer dissatisfaction with service (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.2)

**complete.** (1) &lt;documentation&gt; including all critical information and any necessary, relevant information for the intended audience (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.2)

**complete ICOM code.** (1) diagram feature reference in which dot notation joins an ICOM code to a diagram reference (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.29)

**complete procedure.** (1) all those activities which commence with entry to the procedure and conclude with exit from the procedure (ISO/IEC/IEEE 24765a:2011)

**complete table.** (1) decision table where for all combinations of condition entries there exists a satisfying rule (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.17) Note: In practical terms extended entry tables include limited entries and are therefore mixed entry tables. Any extended or mixed entry table can be transformed into a limited entry table

**completeness.** (1) degree to which an IT service supports all the goals, objectives, and data specified by the user (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.1.1)

**completion code.** (1) code communicated to a job stream processor by a batch program to influence the execution of succeeding steps in the input stream (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**completion criteria.** (1) conditions under which the testing activities are considered complete (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.2)

**completion time theorem.** (1) real-time scheduling theorem (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For a set of independent periodic tasks, if each task meets its first deadline when all tasks start at the same time, the deadlines will be met for any combination of start times.

**complex programmable logic device (CPLD).** (1) hardware component with a fully programmable AND/OR gate array (ISO/IEC/IEEE 24765d:2015)

**complexity.** (1) degree to which a system's design or code is difficult to understand because of numerous components or relationships among components (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to any of a set of structure-based metrics that measure the attribute in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) degree to which a system or component has a design or implementation that is difficult to understand and verify (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also:
simplicity matrix. (1) a table used to allocate a weight to a function type (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: The matrix allocates this weight on the basis of the number of data element types in combination with the number of record types or file types referenced.

complexity of a function. (1) the weight allocated to a function on the basis of which a number of function points is assigned to the function (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

compliance. (1) doing what has been asked or ordered, as required by rule or law (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) (2) continual fulfillment of internal and external requirements, rules, and regulations (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) See also: conformance

component. (1) entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.3) (2) one part that makes up a system (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (3) object that encapsulates its own template, so that the template can be interrogated by interaction with the component (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.26) (4) specific, named collection of features that can be described by an IDL component definition or a corresponding structure in an interface repository (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) (5) functionally or logically distinct part of a system (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (6) object with a discrete information type that is stored in a component content management system, such as a topic, prerequisite, section, image, or video (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.3) (7) product used as a constituent in an assembled product, system or plant (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.4) Note: A component can be hardware or software and can be subdivided into other components. Component refers to a part of a whole, such as a component of a software product or a component of a software identification tag. The terms module, component, and unit are often used interchangeably or defined to be subelements of one another in different ways depending upon the context. The relationship of these terms is not standardized. A component can be independently managed or not from the end-user or administrator's point of view. See also: element, unit

component content management system (CCMS). (1) content management system that supports the entire document- or information-development life cycle from authoring through review and publishing, including the reuse of modular content (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.4) Note: In case the modular content is XML-based, the individual XML elements available for management are defined by the XML schema or DTD. It is not necessary to specify
component dependencies. (1) all the components that are linked to, directly or indirectly, from a single parent component (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.5)

component home. (1) meta-type that acts as a manager for instances of a specified component type (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: Component home interfaces provide operations to manage component life cycles, and optionally, to manage associations between component instances and primary key values.

component implementation. (1) activity of realizing a component, including unit test (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.8)

component integration test. (1) testing of groups of related components (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

component standard. (1) standard that describes the characteristics of data or program components subdivided into other components (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

component testing. (1) testing of individual hardware or software components (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)

component-aware client. (1) client that is defined using the IDL extensions in the component model (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

composite key. (1) key comprising of two or more attributes (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.38) Note: [key style]

composite measure. (1) variable derived from a set of operations of a construct’s multi-item measures defined according to a construct specification (either reflective or formative) that is the way in which the latent variable representing the construct of interest is linked to its measures (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.4)


composite task. (1) task containing nested objects (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

composite type. (1) data type each of whose members is composed of multiple data items (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: atomic type

also: decomposition


computational component. (1) component specified in a computational viewpoint that has a control interface and declared sets of (a) operation interfaces in which it plays a server role (facets), (b) operation interfaces in which it plays a client role (receptacles), (c) operation interfaces originating announcements carrying notifications of typed events (event sources), (d) operation interfaces consuming announcements carrying notifications of typed events (event sinks), (e) operation interfaces supporting accessor and mutators for attributes (attributes) (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.16)

computational container. (1) container for a declared set of computational component types that has a management interface at which it can be requested to create a computational component of one of the types and add it to the container's content, delete a computational component from the container, list the computational components it currently contains, and list the computational factories it provides (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.16)

computational factory. (1) factory that returns an interface reference to the computational object it creates (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.15)

computational interface template. (1) interface template for either a signal interface, a stream interface, or an operation interface (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.10) Note: A computational interface template comprises a signal, a stream or an operation interface signature as appropriate, a behavior specification and an environment contract specification.

computational object template. (1) object template which comprises a set of computational interface templates which the object template can instantiate, a behavior specification, and an environment contract specification (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.9)

computational viewpoint. (1) viewpoint on an ODP system and its environment which enables distribution through functional decomposition of the system into objects which interact at interfaces (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.1.1.3)

computer. (1) functional unit that can perform substantial computations, including numerous arithmetic operations and logic operations without human intervention (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: A computer can consist of a stand-alone unit or several interconnected units. See also: computing device

computer center. (1) facility that includes personnel, hardware, and software, organized to provide information processing services (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: data processing center

computer crime. (1) crime committed through the use, modification, or destruction of hardware, software, or data (ISO/IEC 2382:2015 Information technology -- Vocabulary)

computer generation. (1) category in a historical classification of computers based mainly on the technology used in their manufacture (ISO/IEC 2382:2015 Information technology -- Vocabulary)

computer graphics. (1) methods and techniques for construction, manipulation, storage, and display of images by means of a computer (ISO/IEC 2382:2015 Information technology -- Vocabulary)
computer instruction. (1) statement in a programming language, specifying an operation to be performed by a computer and the addresses or values of the associated operands (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) loosely, any executable statement in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: instruction format, instruction set

computer language. (1) language designed to enable humans to communicate with computers (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: design language, query language, programming language

computer network. (1) data processing nodes and their interconnections for the purpose of data communication (ISO/IEC 2382:2015 Information technology -- Vocabulary)

computer performance evaluation. (1) engineering discipline that measures the performance of computer systems and investigates methods by which that performance can be improved (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: system profile, throughput, utilization, workload model

computer program. (1) combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) syntactic unit that conforms to the rules of a particular programming language and that is composed of declarations and statements or instructions needed for a certain function, task, or problem solution (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: software

computer program abstract. (1) brief description of a computer program that provides sufficient information for potential users to determine the appropriateness of the program to their needs and resources (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

computer resource. (1) element of a data processing system needed to perform required operations (ISO/IEC 2382:2015 Information technology -- Vocabulary)

computer resource allocation. (1) assignment of computer resources to current and waiting jobs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dynamic resource allocation, storage allocation

computer resources. (1) computer equipment, programs, documentation, services, facilities, supplies, and personnel available for a given purpose (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: computer resource allocation

computer science. (1) branch of science and technology that is concerned with information processing by means of computers (ISO/IEC 2382:2015 Information technology -- Vocabulary)

computer software component (CSC). (1) functionally or logically distinct part of a computer software configuration item, typically an aggregate of two or more software units (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: computer software configuration item, software configuration item, software item

computer software configuration item (CSCI). (1) aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: software configuration item (SWCI) See also: computer software component, hardware configuration item, configuration item, software configuration item, software item

computer system. (1) system containing one or more computers and associated software (ISO/IEC/IEEE
system containing one or more components and elements such as computers (hardware), associated software, and data (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.3) Syn: computing system See also: data processing system
computer-aided (CA). (1) pertaining to a technique or process in which a computer does part of the work (ISO/IEC 2382:2015 Information technology -- Vocabulary)
computer-aided design (CAD). (1) use of a computer to design a device or a system, display it on a computer monitor or printer, simulate its operation, and provide statistics on its performance (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The computer is provided with data concerning the item to be designed, how it is to function, and the rules for the way in which the different components can be joined.
computer-aided software engineering (CASE). (1) use of computers to aid in the software engineering process (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.2) Syn: computer aided software engineering See also: integrated development environment
computer-based software system (CBSS). (1) software system running on a computer (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.5) Note: A CBSS can be a data processing system as seen by human users at their terminals or at equivalent machine-user-interfaces. It includes hardware and all software (system software and application software) which is necessary for realizing data processing functions required by its users
computerization. (1) automation by means of computers (ISO/IEC 2382:2015 Information technology -- Vocabulary)
computerize. (1) to automate by means of computers (ISO/IEC 2382:2015 Information technology -- Vocabulary)
computing center. (1) facility designed to provide computer services to a variety of users through the operation of computers and auxiliary hardware and through services provided by the facility's staff (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
computing device. (1) functional unit that can perform substantial computations, including numerous arithmetic operations and logic operations, with or without human intervention (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.6) Note: A computing device can consist of a stand-alone unit, or several interconnected units. It can also be a device that provides a specific set of functions, such as a phone or a personal organizer, or more general functions such as a laptop or desktop computer. See also: computer
COMSEC. (1) communications security (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
concept analysis. (1) derivation of a system concept through the application of analysis (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: analysis
Guidelines for life cycle management, 2.12) Note: It provides the basis for bounding the operating space, system capabilities, interfaces and operating environment. The concept of operations frequently is embodied in long-range strategic plans and annual operational plans.

In the latter case, the concept of operations in the plan covers a series of connected operations to be carried out simultaneously or in succession. The concept is designed to give an overall picture of the organization operations.

Syn: ConOps, CONOPS

See also: operational concept

concept of operations (ConOps) document. (1) user-oriented document that describes a system's operational characteristics from the end user's viewpoint (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) Note: precedes the requirements phase

corrective phase. (1) period of time in the system life cycle during which the user needs are identified and system concepts are described and evaluated (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) Note: precedes the requirements phase

conceptual data model. (1) a data model that illustrates the data groups as they are seen by the user (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)


conceptual system design. (1) system design activity concerned with specifying the logical aspects of the system organization, its processes, and the flow of information through the system (ISO/IEC 2382:2015 Information technology -- Vocabulary)


conciseness. (1) software attributes that provide implementation of a function with a minimum amount of code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

concurrency. (1) property of a system in which events can occur independently of each other, and hence are not ordered (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.6) See also: step

concurrent. (1) 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 (ISO/IEC 24765:2017 Systems and software engineering-
Vocabulary) (2) problem, process, system, or application in which many activities happen in parallel, the order of incoming events is not usually predictable, and events often overlap (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) Note: A concurrent system or application has many threads of control. See also: parallel, simultaneous

current communication diagram. (1) diagram depicting a network of concurrent tasks and their interfaces in the form of asynchronous and synchronous message communication, event synchronization, and access to passive information-hiding objects (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary)

current elaboration. (1) life cycle process instances are enacted concurrently during the project, and the information items and artifacts produced by these process instances evolve concurrently (ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.1) Note: This is referred to as concurrent elaboration of information items.

current enabling (of transition modes). (1) state, for a multiset of transition modes, when all the involved input places contain enough tokens to satisfy the sum of all of the demands imposed on them by each input arc annotation evaluated for each transition mode in the multiset (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.5)

current task architecture. (1) description of the concurrent tasks in a system or subsystem in terms of their interfaces and interconnections (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary)

current. (1) measurable qualitative or quantitative attribute that is stipulated for a requirement and that indicates a circumstance or event under which a requirement applies (ISO/IEC 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.1.5) (2) description of a contingency to be considered in the representation of a problem, or a reference to other procedures to be considered as part of the current (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.6) (3) Boolean expression containing no Boolean operators (ISO/IEC 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.6)

current entry. (1) indication of the relevance of a condition to a particular rule (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.8)

current stub. (1) list of all the conditions to be considered in the description of a problem (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.1)

conditional information. (1) information supplied with every product to which it is relevant (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary)

conditional jump. (1) jump that takes place only when specified conditions are met (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) See also: unconditional jump

conditional process. (1) process that can be mandatory under some specified conditions, can be optional under other specified conditions, and can be out of scope or not applicable under other specified conditions (ISO/IEC TR 29110-5-1-3:2017 Systems and software engineering -- Lifecycle profiles for Very Small Entities (VSEs) -- Part 5-1-3: Software engineering -- Management and engineering guide: Generic profile group -- Intermediate profile, 3.3)

conditional text. (1) text that is marked to be excluded from one or more versions of a final content deliverable
Conduct Procurements. (1) the process of obtaining seller responses, selecting a seller, and awarding a contract

cone of uncertainty. (1) representation of how the uncertainties inherent in a project decrease over the duration of the project (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

confidentiality. (1) degree to which a product or system ensures that data are accessible only to those authorized to have access (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6.1) (2) degree to which an IT service ensures that data are accessible only to those authorized to have access (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.3.1)

configurability. (1) degree of how well a variability mechanism supports the configuration of a member product (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.6)

configuration. (1) arrangement of a computer system or component as defined by the number, nature, and interconnections of its constituent parts (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in configuration management, the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) arrangement of a system or network as defined by the nature, number, and chief characteristics of its functional units (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) requirements, design, and implementation that define a particular version of a system or system component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (5) manner in which the hardware and software of an information processing system are organized and interconnected (ISO/IEC 2382:2015 Information technology -- Vocabulary) (6) collection of objects able to interact at interfaces (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 10.2) See also: configuration item; form, fit, and function; version

configuration audit. (1) in configuration management, an independent examination of the configuration status to compare with the physical configuration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) detailed review of processes, product definition information, documented verification of compliance with requirements, and an inspection of products to confirm that products have achieved their required attributes or conform to released product configuration definition information (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1)

configuration baseline. (1) configuration information formally designated at a specific time during the life of a product, product component, service, or service component (ISO/IEC/IEEE 24765c:2014) Note: Configuration baselines, plus approved changes from those baselines, constitute the current configuration information.

configuration control. (1) an element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: change control, configuration identification, configuration status accounting
configuration control board (CCB). (1) group of people responsible for evaluating and approving or disapproving proposed changes to configuration items, and for ensuring implementation of approved changes (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (2) qualified personnel who evaluate, for approval or disapproval, all proposed changes to the current developmental baseline (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) Note: This authority can approve a proposed change, thus converting it to an approved modification, or can disapprove a proposed change, or can defer a decision. Syn: change authority See also: configuration control board

configuration identification. (1) element of configuration management, consisting of selecting the configuration items for a system and recording their functional and physical characteristics in technical documentation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) current approved technical documentation for a configuration item as set forth in specifications, drawings, associated lists, and documents referenced therein (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control, configuration status accounting

configuration index. (1) document used in configuration management, providing an accounting of the configuration items that make up a product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration item development record, configuration status accounting

configuration item (CI). (1) item or aggregation of hardware, software, or both that is designated for configuration management and treated as a single entity in the configuration management process (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.15) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.13) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.14) (2) component of an infrastructure or an item which is or will be under control of configuration management (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.7) (3) aggregation of work products that is designated for configuration management and treated as a single entity in the configuration management process (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (4) any system element or aggregation of system elements that satisfies an end use function and is designated by the acquirer for separate configuration control (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1) (5) item or aggregation of software that is designed to be managed as a single entity and its underlying components, such as documentation, data structures, scripts (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.12) Note: Configuration items can vary widely in complexity, size and type, ranging from an entire system including all hardware, software and documentation, to a single module or a minor hardware component. CIs have four common characteristics: defined functionality; replaceable as an entity; unique specification; formal control of form, fit, and function. See also: hardware configuration item, computer software configuration item, configuration identification, critical item

configuration item development record. (1) document used in configuration management, describing the development status of a configuration item based on the results of configuration audits and design reviews (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration index, configuration status
configuration management (CM). (1) 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 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

(2) technical and organizational activities, comprising configuration identification, control, status accounting and auditing (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

(3) coordinated activities to direct and control the configuration (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.7) See also: baseline, change management, configuration identification, configuration control, configuration status accounting, configuration audit

configuration management authority. (1) person(s) or group designated to be responsible for assuring that configuration management activities are planned and carried out. (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

configuration management database (CMDB). (1) specific type of repository for CM information, usually a data store, used to record attributes of configuration items, and the relationships between configuration items, throughout their lifecycle. (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

(2) database containing all the relevant details of each configuration item and details of the important relationships between them. (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.8) See also: configuration management system

configuration management plan (CMP). (1) a component of the project management plan that describes how to identify and account for project artifacts under configuration control, and how to record and report changes to them. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

configuration management system. (1) discipline of identifying the components of a continually evolving system to control changes to those components and maintaining integrity and traceability throughout the life cycle (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) a collection of procedures used to track project artifacts and monitor and control changes to these artifacts (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: configuration management database (CMDB)

configuration parameter. (1) parameter provided by variable components or interfaces, so that its value is selected when bindings occur. (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.9)

configuration status accounting (CSA). (1) element of configuration management that consists of the recording and reporting of information needed to manage a configuration effectively. (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.10) Note: This information includes a listing of the approved configuration identification, the status of proposed changes to the configuration, and the implementation status of approved changes. See also: configuration control, configuration identification, configuration index, configuration item, development record.

conflict. (1) change in one version of a file that cannot be reconciled with the version of the file to which it is applied.
Conformance. (1) fulfillment by a product, process or service of specified requirements (2) within the quality management system, a general concept of delivering results that fall within the limits that define acceptable variation for a quality requirement. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: Conformance is used for fulfillment of the requirements of voluntary standards. See also: compliance

Conformance point. (1) reference point at which behavior can be observed for the purposes of conformance testing.

Conformity. (1) fulfillment of a requirement (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.12) Note: In English the word "conformance" is synonymous but deprecated. In French the word "compliance" is synonymous but deprecated.

Conformity assessment. (1) demonstration that specified requirements relating to a product, process, system, person or body are fulfilled.

Conformity assessment body. (1) body that performs conformity assessment services.

Conformity assessment scheme. (1) conformity assessment system as related to specified objects of conformity assessment to which the same particular specified requirements, rules and procedures apply.

Conformity evaluation. (1) systematic examination of the extent to which a product, process, or device fulfills specified requirements.

Conformity evaluation report. (1) document that describes the conduct and results of the evaluation carried out for a Ready to Use software product (RUSP).

Confusion matrix. (1) table used to describe the performance of a classifier on a set of test data for which the true and false values are known.

Connectivity. (1) capability of a system or device to be attached to other systems or devices without modification.

ConOps. (1) concept of operations.

Consecutive. (1) pertaining to the occurrence of two sequential events or items without the intervention of any other
event or item; that is, one immediately after the other (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

consequence. (1) outcome of an event affecting one or more stakeholders (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.1) (2) outcome of an event affecting objectives (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.3) (3) outcome of an occurrence of a particular set of circumstances (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.5) Note: An event can lead to a range of consequences. A consequence can be certain or uncertain and can have positive or negative effects on objectives. Consequences can be expressed qualitatively or quantitatively. Initial consequences can escalate through follow-on effects.

consistency. (1) degree of uniformity, standardization, and freedom from contradiction among the documents or parts of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) software attributes that provide uniform design and implementation techniques and notations (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: traceability

consistent. (1) without internal conflicts (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.3)

consistent state. (1) point at which processing has been fully executed, the Functional User Requirement has been satisfied, and there is nothing more to be done (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.10)

consolidation of an entitlement. (1) process of combining two or more entitlements into a single, unified entitlement (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.5) Note: Entitlements can be consolidated to simplify understanding of the current position or as the result of a licensor negotiation. The entitlement schema enables the recording of entitlement consolidations.

constant. (1) quantity or data item whose value cannot change (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) instance whose identity is known at the time of writing (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.40) (3) specification that an attribute or participant property value, once assigned, shall not be changed, or that an operation shall always provide the same output argument values given the same input argument values (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.40) Note: The identity of a constant state class instance is represented by #K, where K is an integer or a name. See also: variable figurative constant, literal

constant dollar analysis. (1) addressing inflation and deflation by using cash flow amounts that represent money values which are referenced to a fixed time (typically, the beginning of a project) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: actual dollar analysis

constant-failure period. (1) period of time in the life cycle of a system or component during which hardware failures occur at an approximately uniform rate (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: early-failure period, wearout-failure period, bathtub curve
constituent configuration item. (1) individual item to be controlled that is a constituent (part) of a larger configuration item, such as a reference model, hardware prototype or software build (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

constituent system. (1) independent system that forms part of a system of systems (SoS) (ISO/IEC/IEEE 21839:2019 Systems and software engineering--System of systems (SoS) considerations in life cycle stages of a system, 3.1.1) Note: Constituent systems can be part of one or more SoS. Each constituent system is a useful system by itself, having its own development, management, utilization, goals, and resources, but interacts within the SoS to provide the unique capability of the SoS. Syn: CS

constraint. (1) limitation or implied requirement that constrains the design solution or implementation of the systems engineering process and is not changeable by the enterprise (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes) (2) restriction on software life cycle process (SLCP) development (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) a statement that expresses measurable bounds for an element or function of the system (4) rule that specifies a valid condition of data (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.41) (5) a rule that specifies a valid condition of data (6) responsibility that is a statement of facts that are required to be true in order for the constraint to be met (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.41) (7) restriction on the value of an attribute or the existence of any object based on the value or existence of one or more others (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (8) a responsibility that is a statement of facts that are required to be true in order for the constraint to be met (9) externally imposed limitation on system requirements, design, or implementation or on the process used to develop or modify a system (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.6) (10) an externally imposed limitation on system requirements, design, or implementation or on the process used to develop or modify a system (11) a limiting factor that affects the execution of a project, program, portfolio, or process (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (12) externally imposed limitation on the system, its design, or implementation or on the process used to develop or modify a system (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.1.6) Note: A constraint is a factor that is imposed on the solution by force or compulsion and can limit or modify the design changes.

constraints dependency. (1) relationship between variation points, between variants and between a variation point and a variant (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.5)

construct. (1) concept, such as the abstract idea, image, underlying theme, or subject matter, that one wishes to measure using process assessments (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.5) Note: In process measurement frameworks, constructs (also refers to latent constructs) are theoretical concepts such as the process quality characteristics and process attributes. The meaning that one assigns to a construct is its theoretical definition, which should describe its distinct dimensions (facets).

construction. (1) process of writing, assembling, or generating assets (ISO/IEC/IEEE 24765j:2021) (2) activity in


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
software development consisting of detailed design, coding, unit testing, and debugging (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: the collection of activities focused on creating source code

consumable. (1) any part or material that is necessary to be replaced or refilled for continuous use or maintenance of the product (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.6)

customer. (1) individual member of the general public, purchasing or using products for private purposes (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.7) (2) event sink (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: A consumer is assumed to be a nonskilled person. See also: event sink

customer object. (1) object which is a sink of the information conveyed (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.4.4)

customer product. (1) product available to, intended for or likely to be used by consumers (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.8)

customer software package. (1) COTS software product designed and sold for end users to carry out identified functions; the software and its associated documentation are packaged for sale as a unit (ISO/IEC/IEEE 24765a:2011)

customer. (1) object that can act as a factory and can provide the necessary environment for subsequent management of the components created by it (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.27) (2) framework for integrating transactions, security, events and persistence into a component's behavior at runtime (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) (3) model element that owns one or more distinct elements through the special owns(contains) relationships between the container element and owned elements (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) Note: Containers provide the run-time execution environment for CORBA component implementations.

container interface. (1) interface of a data repository allowing access to data (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 14.1.1.2)

content. (1) interactive or non-interactive object containing information represented by text, image, video, sound, or other media (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.6)

content consistency. (1) semantic consistency among the contents of information items (ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.2)

content coupling. (1) type of coupling in which some or all of the contents of one software module are included in the contents of another module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: common-environment coupling, control coupling, data coupling, hybrid coupling, pathological coupling

content management. (1) control of units of information with their metadata, to allow selective reuse in documents or information products with variable structures and formats (ISO/IEC/IEEE 24765h:2019) Note: Content management for
user documentation means management of help topics, explanations of concepts, troubleshooting procedures, compliance statements, and variables such as the names and host platforms of software products, with metadata tags that are applied to format output.

**content management system.** (1) system that supports authoring, storing, translating, and publishing content (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.9) See also: document management system

**content object.** (1) content that is encompassed by an XML elem (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.6)

**content type.** (1) reusable definition of settings for storage, metadata, workflow, and behavior that is assigned to a component (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.7)

**content unit.** (1) identifiable and manageable part of larger information objects (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.8) Note: The individual content units available for management are typically defined by an XML schema or DTD.

**context.** (1) immediate environment in which a function (or set of functions in a diagram) operates (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.30)

**context completeness.** (1) degree to which a product or system can be used with the required levels of effectiveness, efficiency, freedom from risk and satisfaction in each of the specified contexts of use (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.1) Note: Context completeness can be specified or measured either as the degree to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, freedom from risk and satisfaction in all the intended contexts of use, or by the presence of product properties that support use in all the intended contexts of use.

**context coverage.** (1) degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in both specified contexts of use and in contexts beyond those initially explicitly identified (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)-- System and software quality models, 4.1.5) Note: Context of use is relevant to both quality in use and some product quality (sub)characteristics (where it is referred to as specified conditions).

**context diagram.** (1) diagram that presents the context of the top-level function of an IDEF0 model, whose diagram number is a-n, where 0#n#9 (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.31) Note: The one-box A-0 context diagram is a required context diagram; those with diagram numbers A-1, A-2, ..., A-9 are optional context diagrams.

**context diagrams.** (1) a visual depiction of the product scope showing a business system (process, equipment, computer system, etc.), and how people and other systems (actors) interact with it. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: context diagram

**context of use.** (1) users, tasks, equipment (hardware, software and materials), and the physical and social environments
in which a system, product or service is used (ISO/IEC 25063:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description, 3.2) (2) users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.2) Note: Context of use includes direct use or use supported by assistive technologies. Environment includes physical aspects such as equipment and resources as well as social aspects such as demographics and culture.

context-sensitive help. (1) type of on-screen documentation in which the information that is displayed depends upon the user’s view of the software (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.10) See also: embedded documentation, printed documentation

contextual schema. (1) formal description of the boundary of the context of use where data models are applied (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.4) Note: It is a high-level description of the business informational needs. It is more general than a conceptual model, as it includes a holistic vision of a (system) context of the architecture.

contiguous allocation. (1) storage allocation technique in which programs or data to be stored are allocated a block of storage of equal or greater size, so that logically contiguous programs and data are assigned physically contiguous storage locations (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: paging (1)

contingency. (1) an event or occurrence that could affect the execution of the project that may be accounted for with a reserve (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

contingency plan. (1) plan for dealing with a risk factor, if it becomes a problem (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

contingency reserve. (1) time or money allocated in the schedule or cost baseline for known risks with active response strategies (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

contingent response strategies. (1) responses provided which may be used in the event that a specific trigger occurs (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

continual improvement. (1) recurring activity to enhance performance (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.13)

continual process improvement. (1) ongoing cycle of process improvement programs to strengthen and improve the processes supporting business and include one or several improvement projects or initiatives, which can be implemented in series or in parallel (ISO/IEC TR 33014:2013 Information technology--Process assessment--Guide for process improvement, 3.1)

continuity. (1) degree to which the IT service is provided under all foreseeable circumstances, including mitigating the risks resulting from interruption to an acceptable level (ISO/IEC TS 25011:2017 Information technology--Systems and
**continuous delivery.** (1) software engineering practices that allow for frequent releases of new systems (including software) to staging or various test environments through the use of automated tools *(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)*

**continuous deployment.** (1) automated process of deploying changes to production by verifying intended features and validations to reduce risk *(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)* Syn: CD See also: continuous delivery

**continuous forms.** (1) forms produced in continuous lengths during the manufacturing process and intended primarily for use with sprocket-hole transporting mechanisms *(ISO/IEC/IEEE 24765:2020)*

**continuous integration.** (1) technique that continually merges artifacts, including source code updates from all developers on a team, into a shared mainline to build and test the developed system *(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)* Syn: CI

**continuous iteration.** (1) loop that has no exit *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**continuous representation.** (1) capability maturity model structure wherein capability levels provide a recommended order for approaching process improvement within each specified process area *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**continuous risk management.** (1) process of analyzing the progress of a planned activity, project, or program on a periodic, ongoing basis and handling identified risk factors *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* (2) continuous process, which may be automated, that identifies, applies, and monitors controls to treat risks for a planned activity, project, or program, to achieve a desired outcome *(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)* Note: includes developing options and fallback positions to permit alternative solutions to reduce the impact if a risk factor becomes a problem.

**contract.** (1) a mutually binding agreement that obligates the seller to provide the specified product or service or result and obligates the buyer to pay for it *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* (2) agreement governing part of the collective behavior of a set of objects *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.1)* (3) relationship between acquirer and supplier, which in broad terms prescribes that one party will provide defined goods and services and the other party will pay a defined fee for them *(ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.6)* Note: A contract is an agreement between two or more parties regarding a course of action. The formality of a contract can range from a simple informal oral description to a formal written instrument.

**contract administration.** (1) process of managing the contract and the relationship between the acquirer and supplier, including reviewing and documenting how the supplier is performing or has performed; establishing required corrective actions; and managing contract changes *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*
contract change control system. (1) the system used to collect, track, adjudicate, and communicate changes to a contract (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

contract management plan. (1) document that describes how a specific agreement will be administered to monitor delivery of required documentation and performance of the statement of work, to evaluate performance, and to control changes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

contract work breakdown structure (CWBS). (1) portion of the overall work breakdown structure applicable to a contract, developed and maintained by the supplier (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

contractual context. (1) knowledge that a particular contract is in place, and thus that a particular behavior of a set of objects is required (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.2.3)

contractual requirement. (1) result of the analysis and refinement of customer requirements into a set of requirements suitable to be included in one or more solicitation packages, formal contracts, or supplier agreements between the acquirer and other appropriate organizations (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: acquirer, customer requirement

contravariance. (1) rule governing the overriding of a property and requiring that the set of values acceptable for an input argument in the overriding property shall be a superset (includes the same set) of the set of values acceptable for that input argument in the overridden property, and the set of values acceptable for an output argument in the overriding property shall be a subset (includes the same set) of the set of values acceptable for that output argument in the overridden property (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.42)

control. (1) in engineering, the monitoring of system output to compare with expected output and taking corrective action when the actual output does not match the expected output (2) in engineering, the monitoring of system output to compare with expected output and taking corrective action when the actual output does not match the expected output (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) comparing actual performance with planned performance, analyzing variances, assessing trends to effect process improvements, evaluating possible alternatives, and recommending appropriate corrective action as needed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (4) in an IDEF0 model, a condition or set of conditions required for a function to produce correct output (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 2.1.32)

control account. (1) a management control point where scope, budget, actual cost, and schedule are integrated and compared to earned value for performance measurement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: work package

control arrow. (1) arrow or arrow segment that expresses IDEF0 control (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.33) Note: That is, an object type set whose instances establish a condition or set of conditions required for a function to produce correct output. The arrowhead of a control arrow is attached to the top side of a box.
control chart. (1) a graphic display of process data over time and against established control limits, which has a centerline that assists in detecting a trend of plotted values toward either control limit. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

countrol clustering. (1) task-structuring criterion by which a control object is combined into a task with the objects it controls (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

Control Costs. (1) the process of monitoring the status of the project to update the project costs and managing changes to the cost baseline (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

control coupling. (1) type of coupling in which one software module communicates information to another module for the explicit purpose of influencing the latter module’s execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: common-environment coupling, content coupling, data coupling, hybrid coupling, pathological coupling

countrol data. (1) data that select an operating mode, direct the sequential flow of a program, or otherwise directly influence the operation of software (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

countrol field. (1) field comprising one or more input variables whose change in value, or lack of change, between successive logical records affects the flow of control through the main procedure (ISO/IEC/IEEE 24765a:2011)

control flow. (1) sequence in which operations are performed during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) sequence in which operations are performed during the execution of a test item (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.7) Syn: flow of control See also: data flow

countrol flow diagram. (1) diagram that depicts the set of all possible sequences in which operations can be performed during the execution of a system or program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include box diagram, flowchart, input-process-output chart, state diagram. See also: data flow diagram, call graph, structure chart

control flow sub-path. (1) sequence of executable statements within a test item (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.8)

control Information. (1) data coming from outside the application boundary that turns on or off one or more processes of an application or that influences the operation of a transaction (e.g. a signal or selection data) (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) data that influences an elementary process by specifying what, when or how data is to be processed (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.11)

countrol limits. (1) the area composed of three standard deviations on either side of the centerline, or mean, of a normal distribution of data plotted on a control chart, which reflects the expected variation in the data. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: specification limits

control loopback. (1) loopback of output from one function to be control for another function in the same diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.34)
control manager. (1) role that approves or rejects change and manages change-related tasks, such as testing and deployment (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.6) Note: This role may be combined with other roles and is a direct report (or shared role) with the Service Manager. If one person is appointed to the role, the person reports to the Service Manager for service matters and has the authority over change-related tasks. Syn: CM

control procurements. (1) the process of managing procurement relationships, monitoring contract performance, making changes and corrections as appropriate, and closing out contracts (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

control quality. (1) the process of monitoring and recording results of executing the quality activities to assess performance and ensure the project outputs are complete, correct, and meet customer expectations (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: quality control

control resources. (1) the process of ensuring that the physical resources assigned and allocated to the project are available as planned, as well as monitoring the planned versus actual utilization of resources and performing corrective action as necessary (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


Control Schedule. (1) the process of monitoring the status of project activities to update project progress and manage changes to the schedule baseline to achieve the plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

Control Scope. (1) the process of monitoring the status of the project and product scope and managing changes to the scope baseline (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

control statement. (1) program statement that selects among alternative sets of program statements or affects the order in which operations are performed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assignment statement, declaration

control store. (1) in a microprogrammed computer, the computer memory in which microprograms reside (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: microword, nanostore

control task. (1) task that makes decisions to control other tasks' execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

controller. (1) device or computer chip that interfaces with a peripheral device (ISO/IEC/IEEE 24765d:2015)

controller area network (CAN). (1) high-integrity bus system for networking intelligent devices within a system (ISO/IEC/IEEE 24765d:2015) Note: commonly used in embedded networks for vehicles or medical equipment

convention. (1) requirement employed to prescribe a disciplined, uniform approach to providing consistency in a software product, that is, a uniform pattern or form for arranging data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: practice, standard

conversational. (1) pertaining to an interactive system or mode of operation in which the interaction between the user
and the system resembles a human dialog *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

See also: batch, interactive, online, real time

**conversion. (1)** modification of existing software to enable it to operate with similar functional capability in a different environment *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**conversion functionality. (1)** transactional or data functions provided to convert data or provide other user-specified conversion requirements *(ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.12)* Note: Conversion functionality exists only during the development or enhancement of an application.

**convertibility. (1)** the ability to convert the results from applying two or more FSM methods in the measurement of a functional size of the same set of functional user requirements *(ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.3)*

**cookie. (1)** small file that is stored in and retrieved from user web storage to maintain state information, including identification of users and transaction coherency *(ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.7)* Note: Web sites store/retrieve cookies from user client systems to maintain state information including identification of users and transaction coherency.

**COOP. (1)** continuity of operations plan *(ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 3.2)*

**copy. (1)** to read data from a source, leaving the source data unchanged, and to write the same data elsewhere in a physical form that can differ from that of the source *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* result of a copy process as in (1) *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: move

**copyright. (1)** exclusive right granted to the owner of an original work of authorship, which is fixed in any tangible medium of expression, to reproduce, publish, perform, or sell the work *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**COQ. (1)** cost of quality *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*


**core. (1)** processing unit in a computer or processor which manages instructions, data, and operations *(ISO/IEC/IEEE 24765:2014)*

**core report. (1)** document for providing descriptions of the process and outcomes of the benchmarking activity *(ISO/IEC 29155-3:2015 Systems and software engineering--Information technology project performance benchmarking framework--Part 3: Guidance for reporting)* Note: Two kinds of core reports (i.e. executive summary and detailed report) are often produced for reporting results of an instance of benchmarking activity.

**coroutine. (1)** routine that begins execution at the point at which operation was last suspended, and that is not
required to return control to the program or subprogram that called it (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) See also: subroutine

corporate board or equivalent body. (1) person or group of people who assumes legal responsibility for conducting or controlling an organization at the highest level (ISO/IEC 19770-5:2015 Information technology—IT asset management—Overview and vocabulary, 3.9)

corporate governance of IT. (1) at the level of top management, establishment of strategy and policy for the use of IT, and organizational control of the use of IT (ISO/IEC/IEEE 24765c:2014)

correctability. (1) degree of effort required to correct software defects and to cope with user complaints (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary)

corrective action. (1) action to eliminate the cause or reduce the likelihood of recurrence of a detected nonconformity or other undesirable situation (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems—Requirements, 3.14) (2) an intentional activity that realigns the performance of the project work with the project management plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: preventive action

corrective maintenance. (1) reactive modification of a software product performed after delivery to correct discovered problems (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.2) (2) maintenance performed to correct faults in hardware or software (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) Note: The modification repairs the software product to satisfy requirements.

correctness. (1) degree to which a system or component is free from faults in its specification, design, and implementation (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) (2) degree to which software, documentation, or other items meet specified requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) (3) degree to which software, documentation, or other items meet user needs and expectations, whether specified or not (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) (4) degree to which an IT service uses the correct process and produces the correct results with accurate data (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.1.2)


cost aggregation. (1) summing the lower-level cost estimates associated with the various work packages for a given level within the project's WBS or for a given cost control account (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

cost avoidance. (1) revenue (positive cash flow) that results from decreasing expenses, rather than from increasing income (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary)

cost baseline. (1) the approved version of the time-phased project budget, excluding any management reserves, which can be changed only through formal change control procedures and is used as a basis for comparison to actual

cost basis. (1) entire cost to acquire an asset (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: includes the purchase price, delivery, installation, and any other costs to put the asset into service Syn: acquisition cost

cost constraint. (1) limitation or restraint placed on the project budget, such as funds available over time (ISO/IEC/IEEE 24765c:2014)

cost function. (1) objective function that characterizes the cost associated with different values of the decision variable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: income function

cost management plan. (1) a component of a project or program management plan that describes how costs will be planned, structured, and controlled (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

cost of quality (COQ). (1) all costs incurred over the life of the product by investment in preventing non-conformance to requirements, appraisal of the product or service for conformance to requirements, and failure to meet requirements (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: Prevention and appraisal costs (cost of conformance) include costs for quality planning, quality control (QC), and quality assurance to help ensure compliance to requirements (i.e., training, QC systems, etc.). Failure costs (cost of non-conformance) include costs to rework products, components, or processes that are non-compliant, costs of warranty work and waste, and loss of reputation.

cost performance baseline. (1) time-phased budget under change control, used to compare actual expenditures to planned expenditures (ISO/IEC/IEEE 24765c:2014) Note: used to determine if preventive or corrective action is needed to meet the project objectives

cost performance index (CPI). (1) a measure of the cost efficiency of budgeted resources expressed as the ratio of earned value to actual cost (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

cost plus award fee contracts (CPAF). (1) a category of contract that involves payments to the seller for all legitimate actual costs incurred for completed work, plus an award fee representing seller profit (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

cost plus fixed fee contract (CPFF). (1) a type of cost-reimbursable contract where the buyer reimburses the seller for the seller's allowable costs (allowable costs are defined by the contract) plus a fixed amount of profit (fee) (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: cost-plus-fixed-fee contract

cost plus incentive fee contract (CPIF). (1) a type of cost-reimbursable contract where the buyer reimburses the seller for the seller's allowable costs (allowable costs are defined by the contract), and the seller earns its profit if it meets defined performance criteria (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: cost-plus-incentive-fee contract

cost variance (CV). (1) the amount of budget deficit or surplus at a given point in time, expressed as the difference between the earned value and the actual cost (A Guide to the Project Management Body of Knowledge (PMBOK(R))
**cost-benefit analysis.** (1) a financial analysis tool used to determine the benefits provided by a project against its costs (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**cost-plus-fee (CPF).** (1) contract in which the acquirer reimburses the supplier's allowable costs for performing the contract work and also pays a fee (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: cost plus fee

**cost-reimbursable contract.** (1) a type of contract involving payment (reimbursement) by the buyer to the seller for the seller's actual costs, plus a fee typically representing seller's profit (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: cost reimbursable contract


**counter.** (1) variable used to record the number of occurrences of a given event during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**counting rule.** (1) conditions and procedures under which the measurement value is obtained (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**counting scope.** (1) set of Functional User Requirements to be included in the function point count (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.14)


**courtesy.** (1) degree to which the IT service is provided in a polite, respectful and friendly way (ISO/IEC TR 33014:2013 Information technology--Process assessment--Guide for process improvement, 3.2.2.6)

**CPAF.** (1) cost plus award fee (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**CPC.** (1) computer program component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: computer software component

**CPCI.** (1) computer program configuration item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: computer software configuration


**CPF.** (1) cost-plus-fee (ISO/IEC/IEEE 24765c:2014)
CPFF. (1) cost plus fixed fee (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

CPG. (1) clock pulse generator (ISO/IEC/IEEE 24765d:2015)


CPLD. (1) complex programmable logic device (ISO/IEC/IEEE 24765d:2015)

CPPC. (1) cost plus percentage of cost (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

CPU. (1) central processing unit (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)

crash. (1) sudden and complete failure of a computer system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: hard failure

crashing. (1) a technique used to shorten the schedule duration for the least incremental cost by adding resources (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: fast tracking, schedule compression

create WBS (work breakdown structure). (1) the process of subdividing project deliverables and project work into smaller, more manageable components (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

creation. (1) of an &lt;X&gt;, instantiating by an action of objects in the model (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.18) See also: introduction

crisis. (1) critical state of affairs in which a decisive, probably undesirable outcome is impending (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

crisis management. (1) steps to take when a contingency plan does not solve the associated problem (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

criteria. (1) standards, rules, or tests on which a judgment or decision can be based, or by which a product, service, result, or process can be evaluated (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) rules on which a judgment or decision can be based, or by which a product, service, result, or process can be evaluated (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 3.1.6)

critical asset. (1) asset having potential to significantly impact on the achievement of the organization's objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.15) Note: Assets can be safety-critical, environment-critical or performance-critical, and can relate to legal, regulatory or statutory requirements. Critical assets can refer to those assets necessary to provide services to critical customers. Asset systems can be distinguished as being critical in a similar manner to individual assets.

critical chain method. (1) schedule method that allows the project team to place buffers on any project schedule
critical design review (CDR).  (1) review conducted to verify that the detailed design of one or more configuration items satisfy specified requirements; to establish the compatibility among the configuration items and other items of equipment, facilities, software, and personnel; to assess risk areas for each configuration item; and, as applicable, to assess the results of producibility analyses, review preliminary hardware product specifications, evaluate preliminary test planning, and evaluate the adequacy of preliminary operation and support documents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) review as in (1) of any hardware or software component

critical information.  (1) information describing the safe use of the software, the security of the information created with the software, or the protection of the sensitive personal information created by or stored with the software (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.11) (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.6)

critical item. (1) in configuration management, an item within a configuration item that, because of special engineering or logistic considerations, requires an approved specification to establish technical or inventory control at the component level (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

critical path. (1) the sequence of activities that represents the longest path through a project, which determines the shortest possible duration (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: critical path methodology


critical path method (CPM). (1) a method used to estimate the minimum project duration and determine the amount of scheduling flexibility on the logical network paths within the schedule model (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

critical piece first. (1) system development approach in which the most critical aspects of a system are implemented first (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The critical piece can be defined in terms of services provided, degree of risk, difficulty, or other criteria. See also: bottom-up, top-down

critical range. (1) values used to classify software into the categories of acceptable, marginal, or unacceptable (ISO/IEC/IEEE 24765j:2021)

critical section. (1) section of a task's internal logic that is executed mutually exclusively with other tasks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

critical software. (1) software having the potential for serious impact on the users or environment, due to factors including safety, performance, and security (ISO/IEC TR 29110-5-1-4:2018 Software and systems engineering-Lifecycle profiles for very small entities (VSEs)-Part 5-1-4: Software engineering: Management and engineering guidelines: Generic profile group: Advanced profile, 3.7)

critical system. (1) system having the potential for serious impact on the users or environment, due to factors including safety, performance, and security (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very
Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.22) (2) those items (e.g. functions, parts, software, characteristics, processes) having significant effect on the product realization and use of the product -- including safety, performance, form, fit, function, producibility, service life, etc. -- that require specific actions to ensure they are adequately managed (ISO/IEC TR 29110-5-6:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.2)

critical value. (1) value of a validated metric that is used to identify software that has unacceptable quality (ISO/IEC/IEEE 24765:2021)

criticality. (1) degree of impact that a requirement, module, error, fault, failure, or other item has on the development or operation of a system (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)

cross-assembler. (1) assembler that executes on one computer but generates machine code for a different computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

cross-compiler. (1) compiler that executes on one computer but generates machine code for a different computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

cross-reference generator. (1) software tool that accepts as input the source code of a computer program and produces as output a listing that identifies each of the program’s variables, labels, and other identifiers and indicates which statements in the program define, set, or use each one (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: cross-referencer

cross-reference list. (1) list that identifies each of the variables, labels, and other identifiers in a computer program and indicates which statements in the program define, set, or use each one (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

cross-reference tool. (1) software maintenance tool that lets the user determine where a variable is used or where a particular procedure is called on (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

CRS. (1) commonality and reuse strategy (ISO/IEC 26554:2018 Information technology--Software and systems engineering-Tools and methods for product line testing, 4)


cryptographic hash. (1) Method to verify the authenticity of a system element or software via the production of a checksum (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)


CSC. (1) computer software component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

CSCI. (1) computer software configuration item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: SWCI
CSF. (1) critical success factor (ISO/IEC TR 14471:2007 Information technology--Software engineering--Guidelines for the adoption of CASE tools, 2.2)


CSS2. (1) cascading stylesheets level 2 (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.2.2)

CT. (1) communication technology (ISO/IEC/IEEE 24765:2014)

cumulative flow diagram (CFD). (1) a chart indicating features completed over time, plus features in development, and those features in the backlog (Software Extension to the PMBOK(R) Guide Fifth Edition) Note: may indicate features at some intermediate milestones, such as features designed but not yet constructed

curriculum standard. (1) standard that describes the characteristics of a course of study on a body of knowledge that is offered by an educational institution (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

CUS. (1) customer (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.7)

custom software. (1) software product developed for a specific application from a user requirements specification (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.3)

customer. (1) organization or person that receives a product or service (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.16) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.14) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 3.15) (2) an individual or organization who specifies the requirements for and formally accepts delivery of a new or modified hardware or software product and its documentation (3) organization or part of an organization that receives a service or services (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.3) (4) organization or part of an organization that receives a service or services or products of the application management organization (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.14) (5) the entity or entities for whom the requirements are to be satisfied in the system being defined and developed (6) person or organization that could or does receive a product or a service that is intended for or required by this person or organization (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.22) (7) relationship with the supplier of an organization or person that receives or uses a product or service (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.3) (8) individual or organization that purchases or receives a product (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.9) Note: The term "customer" includes but has a broader meaning than "consumer". A customer...
can be internal or external to the organization. This is the entity to whom the system developer must provide proof that the system developed satisfies the system requirements specified. Customers are a subset of stakeholders.

An application management organization can have customers that are internal or external business information management organizations and other application management organizations.

A user or end user is a person that actually uses the application software, where a customer is a person or organization that decides about and acquires the products or services. The customer or user organization is, in its relationships with application management, represented by business information management. Syn: beneficiary, CUS, purchaser See also: acquirer, buyer, stakeholder, user

customer requirement. (1) result of eliciting, consolidating, and resolving conflicts among the needs, expectations, constraints, and interfaces of the product's relevant stakeholders in a way that is acceptable to the customer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

customer satisfaction. (1) state of fulfillment in which the needs of a customer are met or exceeded for the customer's expected experiences as assessed by the customer at the moment of evaluation (ISO/IEC/IEEE 24765h:2019)

customizability. (1) degree to which the IT service can be customized at the request of users (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.7.1)

customization. (1) adaptation of a software or information product to the needs of a particular audience (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.7) (2) modification of a document type definition to add new structures or change the document type definition in a way that is not compatible with a previous structure (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.10)

cut-off date. (1) date after which changes to the software are reflected in the next, rather than the current, software release or issue of the documentation (ISO/IEC/IEEE 24765a:2011)

cutover. (1) transfer of functions of a system to its successor at a given moment (ISO/IEC 2382:2015 Information technology -- Vocabulary)


CVE. (1) Common Vulnerabilities and Exposures (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

CVSS. (1) Common Vulnerability Scoring System (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

CWBS. (1) contract work breakdown structure (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

cycle. (1) period of time during which a set of events is completed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) set of operations that is repeated regularly in the same sequence, possibly with variations in each repetition (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software
development cycle, software life cycle

cycle stealing. (1) process of suspending the operation of a central processing unit for one or more cycles to permit the occurrence of other operations, such as transferring data from main memory in response to an output request from an input/output controller (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

cycle time. (1) time associated with one complete operation of a repetitive process (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.10)

cyclic search. (1) storage allocation technique in which each search for a suitable block of storage begins with the block following the one last allocated (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

danger. (1) hazardous situation, which if not avoided, can result in death or serious injury (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.8)

dangerous condition. (1) state of a system which, in combination with some states of the environment, will result in adverse consequence (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.11) (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.4) Note: A hazardous situation in ISO/IEC Guide 51 and IEC 61508-4 is an instance of a dangerous condition. A concept of dangerous conditions is introduced in order to cover not only hazardous situations in the safety context but also errors in the reliability, integrity, confidentiality, or dependability contexts and other states of a system which can lead to adverse consequences. Occurrences of failures in the context of reliability often, but not always, lead to dangerous conditions. A dangerous condition therefore has attributes, at least, a) the associated adverse consequences, b) the trigger events that lead to the dangerous condition, and c) the trigger events that lead to the adverse consequences from the dangerous condition.

dark matter. (1) the work missed in the original project plan that is required to complete the deliverable product (Software Extension to the PMBOK(R) Guide Fifth Edition)

Darwin Information Typing Architecture (DITA). (1) XML-based architecture for authoring, producing, and delivering topic-oriented, information-typed content that can be reused and single-sourced in a variety of ways (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.13)

Data may be a captured, measured, or recorded representation of information, before it is analyzed, interpreted, or processed. Data may relate to objects such as facts, events, things, processes, or ideas, including concepts that within a certain context have a particular meaning. See also: data type, information

**Data abstraction.** (1) process of extracting the essential characteristics of data by defining data types and their associated functional characteristics and disregarding representation details (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) the result of the process in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: encapsulation, information hiding

**Data analysis.** (1) systematic investigation of the data and their flow in a real or planned system (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**Data analysis technique.** (1) technique used to organize, assess, and evaluate data and information (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**Data attribute.** (1) smallest parcel of information, within an identified data group, carrying a meaning from the perspective of the software’s functional user requirements (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 3.4)

**Data bank.** (1) set of data related to a given subject and organized in such a way that it can be consulted by subscribers (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**Data breakpoint.** (1) breakpoint that is initiated when a specified data item is accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: storage breakpoint See also: code breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint

**Data buffer register (DBR).** (1) region of a physical memory storage used to temporarily store data while it is being moved, e.g., from input to processing (ISO/IEC/IEEE 24765d:2015)

**Data characteristic.** (1) inherent, possibly accidental, trait, quality, or property of data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**Data communication.** (1) transfer of data among functional units according to sets of rules governing data transmission and the coordination of the exchange (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**Data coupling.** (1) type of coupling in which output from one software module serves as input to another module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: input-output coupling See also: common-environment coupling, content coupling, control coupling, hybrid coupling, pathological coupling

**Data date.** (1) a point in time when the status of the project is recorded (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: as-of date, time-now date

**Data declaration source statement.** (1) source statement that reserves or initializes memory at compilation time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**Data definition.** (1) statement where a variable is assigned a value (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.9) Syn: variable definition

**Data definition c-use pair.** (1) data definition and subsequent computation data use, where the data use uses the value defined in the data definition (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing --
data definition p-use pair. (1) data definition and subsequent predicate data use, where the data use uses the value defined in the data definition (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.10)

data definition-use pair. (1) data definition and subsequent data use, where the data use uses the value defined in the data definition (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.11)

data dictionary. (1) collection of information about data such as name, description, creator, owner, provenance, translation in different languages, and usage (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.7)

data element. (1) unique, user-recognizable, non-repeated field in a BFC (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size measurement method, 3.3) (2) smallest unit of data of an IT project (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 3.11) Note: A data element can be a character string, or a digital or graphical element in a BFC. When 'data elements' are indicated for a BFC, the number of data elements is always greater than 0. Syn: data item

data element type (DET). (1) unique, user-recognizable, non-repeated attribute (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.15) (2) unique, non-repeated attribute, which can be in business data, reference data, or code data (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) (3) the most elementary form of data as seen by the user that serves for controlling, recording, or transferring information (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (4) a unique, user-recognizable, non-recursive item of information (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

data exception. (1) exception that occurs when a program attempts to use or access data incorrectly (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: addressing exception, operation exception, overflow exception, protection exception, underflow exception

data file. (1) set of related data records treated as a unit (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.7)

data flow. (1) sequence in which data transfer, use, and transformation are performed during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: control flow

data flow diagram (DFD). (1) diagram that depicts data sources, data sinks, data storage, and processes performed on data as nodes, and logical flow of data as links between the nodes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: data flowchart, data flow graph See also: control flow diagram, data structure diagram

data format. (1) specified arrangement and encoding for data to be communicated or stored and retrieved (ISO/IEC/IEEE 24765d:2015) (2) arrangement of data for storage or display (ISO/IEC 25024:2015 Systems and software engineering-Vocabulary)
data function. (1) functionality provided to the user to meet internal or external data storage requirements (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.16) (2) logical file that is a logical group of permanent data seen from the perspective of the user (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: FPA assigns each data function a type and distinguishes between the following types: the internal logical file and the external interface file.

data function type. (1) one of two categories that FPA assigns to a data function; internal logical file and external interface file (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

data gathering technique. (1) technique used to collect data and information from a variety of sources (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

data group. (1) a distinct, non-empty, non-ordered and non-redundant set of data elements (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size measurement method, A.5) (2) a distinct, non empty, non ordered and non redundant set of data attributes where each included data attribute describes a complementary aspect of the same object of interest (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method; Part 1: Definition of concepts) Note: Each included data element describes a complementary aspect of the same object of interest. A data group is characterized by its persistence. Syn: data group type See also: object of interest

data information. (1) data that enters or exits the application and that satisfies the user's information need (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

data input sheet. (1) user documentation that describes, in a worksheet format, the required and optional input data for a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: user manual

data inventory. (1) in an information processing system, all the data and their characteristics, including interdependencies (ISO/IEC 2382:2015 Information technology -- Vocabulary)

data item. (1) smallest identifiable unit of data within a certain context for which the definition, identification, permissible values, and other information is specified by means of a set of properties (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.9) Note: Data item is a physical object ‘container’ of data values. Syn: field

data management. (1) in a data processing system, the functions that provide access to data, perform or monitor the storage of data, and control input-output operations (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) disciplined processes and systems that plan for, acquire, and provide stewardship for business and technical data, consistent with data requirements, throughout the data lifecycle (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
data manipulation. (1) any processing of the data other than a movement of the data into or out of a functional process, or between a functional process and persistent storage (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.6)

data medium. (1) material in or on which data can be recorded and from which data can be retrieved. Plural: data media (ISO/IEC 2382:2015 Information technology -- Vocabulary)

data model. (1) graphical and textual representation of analysis that identifies the data needed by an organization to achieve its mission, functions, objectives, and strategies and to manage and rate the organization (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.44) (2) model about data by which an interpretation of the data can be obtained in the modeling tool industry (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (3) graphical or lexical representation of data, specifying their properties, structures, and interrelationships (ISO/IEC 16350-2015 Information technology-- Systems and software engineering--Application management, 4.15) Note: A data model can be encoded and manipulated by a computer. A data model identifies the entities, domains (attributes), and relationships (associations) with other data and provides the conceptual view of the data and the relationships among data [key style]. A distinction is made between a logical (or functional) and a technical data model. A logical data model is a representation of an organization's data, organized in terms of entities and relationships and is independent of any particular data management technology. In a technical data model, it is determined in what form data are recorded in the database and in which way the data are approached.

data movement (-type). (1) base functional component which moves a single data group (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 3.6) Note: The COSMIC Functional Size Measurement Method has four types of data movements: Entry, Exit, Read and Write. For measurement purposes, each data movement is considered to account for certain associated data manipulation

data processing (DP). (1) systematic performance of operations upon data (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: The term data processing is not a synonym for information processing. Syn: automatic data processing (ADP)

data processing system. (1) one or more computers, peripheral equipment, and software that perform data processing (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: computer system, computing system

data protection. (1) implementation of appropriate administrative, technical, or physical means to guard against unauthorized intentional or accidental disclosure, modification, or destruction of data (ISO/IEC 16350-2015 Information technology -- Vocabulary)

data provider. (1) individual or organization that is a source of data (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.5)

data quality. (1) degree to which the characteristics of data satisfy stated and implied needs when used under specified conditions (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.5) (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.1)

data quality characteristic. (1) category of data quality attributes that bears on data quality (ISO/IEC 25024:2015 Systems and software engineering--Quality measures, 3.8)
data quality measure. (1) variable to which a value is assigned as the result of measurement of a data quality characteristic (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.12)

data quality model. (1) defined set of characteristics which provides a framework for specifying data quality requirements and evaluating data quality (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.13)

data record. (1) defined group of related data elements, in which all the necessary data elements are included to represent attributes of interest (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 3.12)

data repository. (1) object providing the storage function (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 14.1.1.1)

data representation technique. (1) graphic representation or other method used to convey data and information (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

data store. (1) organized and persistent collection of data and information that allows for its retrieval (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.6)

data structure. (1) physical or logical relationship among data elements, designed to support specific data manipulation functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The data structures are usually documented in technical and logical data models.

data structure diagram. (1) diagram that depicts a set of data elements, their attributes, and the logical relationships among them (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data flow diagram, entity-relationship diagram

data structure-centered design. (1) software design technique in which the architecture of a system is derived from analysis of the structure of the data sets with which the system must deal (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structure clash, structured design, transaction analysis, transform analysis

data submitter. (1) person or organization that provides IT project data to be included into a benchmarking repository (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 3.1)

data transfer controller (DTC). (1) functional unit to control data communication without going through the central processing unit (CPU) (ISO/IEC/IEEE 24765e:2015)

data type. (1) class of data, characterized by the members of the class and the operations that can be applied to them (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) categorization of an abstract set of possible values, characteristics, and set of operations for an attribute (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.45) (3) set of values and operations on those values (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and
Semantics for IDEF1X97 (IDEFobject), 3.1.45) (4) categorization of values operation arguments, typically covering both behavior and representation (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.4) Note: [key style]

data use. (1) executable statement where the value of a variable is accessed (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.13)

data value. (1) content of data item (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.17) Note: Data quality refers to data itself, such as data domain values and possible restrictions.


data-sensitive fault. (1) fault that causes a failure in response to some particular pattern of data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: pattern-sensitive fault See also: program-sensitive fault

data-structure-oriented design. (1) design methodology used for business applications by basing the design on the logical data structures of the program specification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

database. (1) collection of interrelated data stored together in one or more computerized files (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) collection of data organized according to a conceptual structure describing the characteristics of the data and the relationships among their corresponding entities, supporting one or more application areas (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 3.1.8) (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) collection of data describing a specific target area that is used and updated by one or more applications (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size measurement method, A.4) Syn: data base

database design specification. (1) document that describes the content and format of the permanent or semi-permanent data necessary for the software to carry out its functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

database management system. (1) organized collection of structured data (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.18) Note: In order to use database management systems (DBMS), it is necessary to represent data and the relative operations on it in terms of a data model, a data definition and manipulation language

database view. (1) result set of a stored query on the data, which the database users can query just as they would in a persistent database collection object (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: This pre-established query command is kept in the database dictionary. Unlike ordinary base tables in a relational database, it is a virtual table computed or collated dynamically from data in the database, when access to that view is requested. Changes applied to the data in a relevant underlying table are reflected in the data shown in subsequent invocations of the view.

DataType. (1) set of possible values or structures of data (IEEE 1175.3-2004 IEEE Standard for CASE Tool
Interconnections - Reference Model for Specifying Software Behavior

datum. (1) singular of "data" (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: "Data" is commonly used for both singular and plural.

DBMS. (1) Database Management System (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 5)

DBR. (1) data buffer register (ISO/IEC/IEEE 24765d:2015)


DDR2. (1) double data rate x 2 (twice as fast as DDR) (ISO/IEC/IEEE 24765c:2014)

DDR2 SDRAM. (1) double data rate synchronous dynamic random access memory unit with higher performance than DDR SDRAM, because the device transfers data four times (four consecutive words) in one internal clock cycle. (ISO/IEC/IEEE 24765c:2014)

DDR3. (1) double data rate 3, which transfers data 2 to the third power (8 times) faster than DDR (ISO/IEC/IEEE 24765c:2014)

DDR3 SDRAM. (1) double data rate synchronous dynamic random access memory unit with higher performance because it transfers data 2 to the third power (8 times) (8 consecutive words) in one internal clock cycle. (ISO/IEC/IEEE 24765c:2014)

DDR4. (1) double data rate 4; data transfer is 2 to the 4th power = 16 times that of a SDRAM (ISO/IEC/IEEE 24765c:2014)

DDR4 SDRAM. (1) double data rate synchronous dynamic random access memory unit with higher performance, because it transfers data at the rate of 2 to the 4th power (16) times (16 consecutive words) in one internal clock cycle (ISO/IEC/IEEE 24765c:2014)

deactivation. (1) checkpointing a cluster, followed by deletion of the cluster (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.23)

deadlock. (1) situation in which computer processing is suspended because two or more devices or processes are each awaiting resources assigned to the others (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) situation in which two or more tasks are suspended indefinitely because each task is waiting for a resource acquired by another task (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: lockout
deblock. (1) to separate the parts of a block (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: block (2)
debug. (1) to detect, locate, and correct faults in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to detect, locate, and eliminate errors in programs (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: Techniques include use of breakpoints, desk checking, dumps, inspection, reversible execution, single-step operation, and traces.
decision. (1) types of statements in which a choice between two or more possible outcomes controls which set of
actions will result (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.8) Note: Typical decisions are simple selections (e.g. if-then-else), to decide when to exit loops (e.g. while-loop), and in case (switch) statements (e.g. case-1-2-3-...-N).

decision criteria. (1) thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.7)

decision gate. (1) approval event (INCOSE Systems Engineering Handbook, 4th ed.) Note: often associated with a review meeting. Entry and exit criteria are established for each decision gate; continuation beyond the decision gate is contingent on the agreement of decision-makers

decision outcome. (1) result of a decision (which therefore determines the control flow alternative taken) (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.14)

decision rule. (1) combination of conditions (also known as causes) and actions (also known as effects) that produce a specific outcome in decision table testing and cause-effect graphing (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.15)

decision table. (1) table of all contingencies that are to be considered in the description of a problem together with the action to be taken (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.1) (2) table used to show sets of conditions and the actions resulting from them (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) table of conditions that are to be considered in the analysis of a problem, together with the action to be taken for each condition (ISO/IEC 2382:2015 Information technology -- Vocabulary) (4) table that specifies decision variables (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.8)

decision tree. (1) supervised-learning model for which inference can be represented by traversing one or more tree-like structures (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.25)


decision variable. (1) representation of different values for a decision which the decision-maker can choose (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: for example, in an economic life calculation, the decision variable is how long to keep the asset

decision-making technique. (1) technique used to select a course of action from different alternatives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

declaration. (1) action that establishes a state of affairs in the environment of the object making the declaration (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.5) (2) non-executable program statement that affects the assembler or compiler's interpretation of other statements in the program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) set of statements which define the sets, constants, parameter values, typed variables, and functions required for defining the annotations on a
high-level Petri net (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.7) Note: The essence of a declaration is that, by virtue of the act of declaration itself and the authority of the object or its principal, it causes a state of affairs to come into existence outside the object making the declaration.

declarative language. (1) nonprocedural language that permits the user to declare a set of facts and to express queries or problems that use these facts (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: interactive language, rule-based language
decompile. (1) to translate a compiled computer program from its machine language version into a form that resembles, but is not necessarily identical to, the original high-order language program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: compile
decompiler. (1) software tool that decompiles computer programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
decoupling. (1) process of making software modules more independent of one another to decrease the impact of changes to, and errors in, the individual modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: coupling
deep learning. (1) approach to creating rich hierarchical representations through the training of neural networks with one or more hidden layers (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.26) Note: Deep learning uses multi-layered networks of simple computing units (or neurons). In these neural networks each unit combines a set of input values to produce an output value, which in turn is passed on to other neurons downstream.
defect. (1) imperfection or deficiency in a work product where that work product does not meet its requirements or specifications and needs to be either repaired or replaced (ISO/IEC 23531:2020, Systems and software engineering Capabilities of issue management tools, 3.1) (2) an imperfection or deficiency in a project component where that component does not meet its requirements or specifications and needs to be either repaired or replaced (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) generic term that can refer to either a
fault (cause) or a failure (effect)  (*IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, 2.1*) *(4)* fault or deviation from the intended level of performance of a system or software (*ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.4*) *(5)* imperfection or deficiency in a work product or characteristic that does not meet its requirements or specifications (*IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1*) See also: fault
defect density. *(1)* number of defects per unit of product size (*ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary*)
defect repair. *(1)* an intentional activity to modify a non-conforming product or product component (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)
defensive programming. *(1)* a general approach to programming that assumes that errors will occur during both initial development and maintenance and, as a result, creates code in such a way that the program still operates properly when errors occur (*ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary*)
Define Activities. *(1)* the process of identifying the specific actions to be performed to produce the project deliverables (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)
Define Scope. *(1)* the process of developing a detailed description of the project and product (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)
defined process. *(1)* implemented process that is managed and tailored from the organization’s set of standard processes according to the organization’s tailoring guidelines (*ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.2*) Note: A defined process has a process description that is documented and maintained and contributes work products, measures, and other process improvement information to the organization’s process assets. A project’s defined process provides a basis for planning, performing, and improving the tasks and activities of the project.
definition-use pair. *(1)* data definition and subsequent predicate or computational data use, where the data use uses the value defined in the data definition (*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.16*)
definition-use path. *(1)* data definition and subsequent predicate or computational data use, where the data use uses the value defined in the data definition (*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.16*)
definitive software library (DSL). *(1)* secure storage environment, formed of physical media or of one or more electronic software repositories, capable of control and protection of definitive authorized versions of all software configuration items and masters of all software controlled by SAM (*ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.11*)
degree of influence (DI). *(1)* a numerical indicator of the impact of each of the 19 (or more) technical complexity adjustment factors, ranging from 0 (no influence) to 5 (strong influence, throughout) (*ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10*) Note: These indicators are used to compute the value adjustment factor
delegation. (1) action that assigns something, such as authorization, responsibility or provision of a service, to another object (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.6) Note: A delegation, once made, can later be withdrawn.

deleated source statement. (1) source statement that is removed or modified from an existing software product as a new product is constructed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


delimiter. (1) character or set of characters used to denote the beginning or end of a group of related bits, characters, words, or statements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

deliver primitive. (1) service primitive for which the protocol object is the responding object of the corresponding communication (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.6)

deliverable. (1) any unique and verifiable product, result, or capability to perform a service that must be produced to complete a process, phase, or project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) item to be provided to an acquirer or other designated recipient as specified in an agreement (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: This item can be a document, hardware item, software item, service, or any type of work product. See also: acquirer, product, result

deliverable product. (1) unique and verifiable system or software product to perform a service, that is subject to approval by the project sponsor or customer (ISO/IEC 25041: 2012 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation guide for developers, acquirers and independent evaluators, 4.1)

delivered source statement. (1) source statement that is incorporated into the product delivered to the customer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

delivery. (1) release of a system or component to its customer or intended user (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software life cycle, system life cycle

delphi technique. (1) information-gathering technique used as a way to reach a consensus of experts on a subject (ISO/IEC/IEEE 24765h:2019) Note: Experts on the subject participate in this technique anonymously. A facilitator uses a questionnaire to solicit ideas about the important project points related to the subject. The responses are summarized and are then recirculated to the experts for further comment. Consensus may be reached in a few rounds of this process. The Delphi technique helps reduce bias in the data and keeps any one person from having undue influence on the outcome. Syn: Delphi technique

delta. (1) difference between two versions (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.11)

demand paging. (1) storage allocation technique in which pages are transferred from auxiliary storage to main storage only when those pages are needed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: anticipatory paging

DEMIL. (1) demilitarization (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense
**demodularization.** (1) in software design, the process of combining related software modules, usually to optimize system performance (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: downward compression, lateral compression, upward compression

demonstration. (1) dynamic analysis technique that relies on observation of system or component behavior during execution, without need for post-execution analysis, to detect errors, violations of development standards, and other problems (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: testing

dependability. (1) trustworthiness of a computer system such that reliance can be justifiably placed on the service it delivers (IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, 2.2) (2) of an item, ability to perform as and when required (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.1.7)

dependency export. (1) operation in which a component and all of its dependencies are exported from the CCMS as a single process (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.11)

dependent entity. (1) entity for which the unique identification of an instance depends upon its relationship to another entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.46) Note: Expressed in terms of the foreign key, an entity is said to be dependent if any foreign key is wholly contained in its primary key. Syn: identifier-dependent entity See also: independent entity [key style]

dependent state class. (1) class whose instances are, by their very nature, intrinsically related to certain other state class instance(s) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.47) Note: It would not be appropriate to have a dependent state class instance by itself and unrelated to an instance of another class(es) and, furthermore, it makes no sense to change the instance(s) to which it relates. See also: independent state class

deployment. (1) stage of a project in which a system is put into operation and transition issues are resolved (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) (2) of requirements assignment of requirements along with the system decomposition (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.3) See also: release

deployment package. (1) set of artifacts developed to facilitate the implementation of a set of practices of the selected framework in a very small entity (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.8) Syn: DP

derivation of requirements. (1) translation and elaboration of requirements from one type of requirements to another in the same system level (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.4) Note: Types of requirements include quality in use requirements, product quality requirements, and data requirements.

derived class. (1) relation between a template class CA of instances of A, and template class CB of instances of B, where template A is an incremental modification of template B (ISO/IEC 10746-2:2009 Information technology -- Open Systems Interconnection -- Information and data model -- Part 2: General concepts and mechanisms, 9.2.2) Syn: sub-class


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.

Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
distributed processing -- reference model: foundations, 9.24) see also: base class

**derived data.** (1) data created as a result of processing that involves steps other than or in addition to direct retrieval and validation of information from data functions (ISO/IEC 20926:2009 software and systems engineering -- software measurement -- IFPUG functional size measurement method 2009, 3.17) (ISO/IEC 24570:2018 software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)

**derived measure.** (1) measure that is defined as a function of two or more values of base measures (ISO/IEC/IEEE 15939:2017 systems and software engineering--Measurement process, 3.8) Note: A transformation of a base measure using a mathematical function can also be considered as a derived measure.

**derived property.** (1) designation given to a property whose value is determined by computation (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.50) Note: The typical case of a derived property is as a derived attribute although there is nothing to prohibit other kinds of derived property. Syn: derived attribute, derived participant party

**derived requirement.** (1) requirement deduced or inferred from the collection and organization of requirements into a particular system configuration and solution (ISO/IEC/IEEE 29148:2018 systems and software engineering-Life cycle processes-Requirements engineering, 4.1.8) (2) a requirement deduced or inferred from the collection and organization of requirements into a particular system configuration and solution (3) requirement that is not explicitly stated in customer requirements, but is inferred from contextual requirements (such as applicable standards, laws, policies, common practices, and management decisions) or from requirements needed to specify a product or service component (ISO/IEC/IEEE 24765:2017 systems and software engineering-Vocabulary) (4) detailed characteristic of the system of interest (SOI) that typically is identified during elicitation of stakeholder requirements, requirements analysis, trade studies, or validation (INCOSE Systems Engineering Handbook, 4th ed.) Note: Derived requirements can arise during analysis and design of components of the product or service. See also: product requirement

**derived type.** (1) data type whose members and operations are taken from those of another data type according to some specified rule (ISO/IEC/IEEE 24765:2017 systems and software engineering-Vocabulary) See also: subtype


**descendent diagram.** (1) decomposition diagram related to a specific box by a hierarchically consecutive sequence of one or more child/parent relationships (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.38)

**description.** (1) information item that represents a planned or actual concept, function, design, or object (ISO/IEC/IEEE 15289:2019 systems and software engineering--Content of life-cycle information items (documentation), 5.8)

**description standard.** (1) standard that describes the characteristics of product information or procedures provided to help understand, test, install, operate, or maintain the product (ISO/IEC/IEEE 24765:2017 systems and software engineering-Vocabulary)

**design.** (1) [process] to define the architecture, system elements, interfaces, and other characteristics of a system or
design attribute. (1) element of a design view that names a characteristic or property of a design entity, design relationship, or design constraint (ISO/IEC/IEEE 24765:2021) See also: design constraint, design entity, design relationship design authority. (1) person or organization that is responsible for the design of the product (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.5.3) (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.5) design characteristic. (1) design attributes or distinguishing features that pertain to a measurable description of a product or service (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.19) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.17) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.18) design concept. (1) fundamental idea that can be applied to designing a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) design constraint. (1) explicit and direct restriction regarding the choice of design ideas (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) (2) boundary condition, externally or internally imposed, for the system of interest, within which the organization remains when executing the processes during the concept and development stages (INCOSE Systems Engineering Handbook, 4th ed.) Note: It either declares a design idea to be
compulsory or to be excluded. See also: design attribute, design entity, design relationship
design description. (1) document that describes the design of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Typical contents include system or component architecture, control logic, data structures, input/output formats, interface descriptions, and algorithm. Syn: design document, design specification See also: product specification, requirements specification
design fault. (1) design (specification, coding) fault that results from a human error during system design and that might result in a design failure (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
design language. (1) specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a hardware or software design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) standardized notation, modeling technique, or other representation scheme and its usage conventions, shown to be effective in representing and communicating design information (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include hardware design language, program design language. See also: requirements specification language
design level. (1) design decomposition of the software item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
design methodology. (1) systematic approach to creating a design consisting of the ordered application of a specific collection of tools, techniques, and guidelines (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
design pattern. (1) description of the problem and the essence of its solution to enable the solution to be reused in different settings (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: not a detailed specification, but a description of accumulated wisdom and experience.
design phase. (1) the period in the software life cycle during which definitions for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) the period in the software life cycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: detailed design, preliminary design
design rationale. (1) information capturing the reasoning of the designer that led to the system as designed, including design options, trade-offs considered, decisions made, and the justifications of those decisions (ISO/IEC/IEEE 24765:2021)
design requirement. (1) requirement that specifies or constrains the design of a system or system component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement
design review. (1) formal, documented, comprehensive, and systematic examination of a design to determine if the design meets the applicable requirements, to identify problems, and to propose solutions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process or meeting during which a system, hardware, or software design is presented to project personnel, managers, users, customers, or other interested parties for comment or approval (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include critical design review,
preliminary design review, system design review. See also: code review, formal qualification review, requirements review, test readiness review

design standard. (1) standard that describes the characteristics of a design or a design description of data or program components (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

design strategy. (1) overall plan and direction for performing design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

design unit. (1) logically related collection of design elements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: In an Ada PDL, a design unit is represented by an Ada compilation unit.

design view. (1) representation comprised of one or more design elements to address a set of design concerns from a specified design viewpoint (ISO/IEC/IEEE 24765j:2021) See also: design concern, design element, design viewpoint

design viewpoint. (1) specification of the elements and conventions available for constructing and using a design view (ISO/IEC/IEEE 24765j:2021) See also: design view

design-to-cost. (1) approach to managing a system or software project so as to hold the project to a predetermined cost (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Actual and projected costs are closely tracked, and actions such as deleting or postponing lower-priority requirements are taken if costs threaten to exceed targets. Syn: cost as an independent variable (CAIV), design to cost

desirable consequence. (1) consequence associated with a benefit or gain or avoiding an adverse consequence (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.4) Syn: positive consequence

desk checking. (1) manual simulation of program execution to detect faults through step-by-step examination of the source program for errors in function or syntax (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) static analysis technique in which code listings, test results, or other documentation are visually examined, usually by the person who generated them, to identify errors, violations of development standards, or other problems (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: inspection, walk-through

desktop publishing. (1) electronic publishing using a microcomputer (ISO/IEC 2382:2015 Information technology -- Vocabulary)

destination address. (1) address of the device or storage location to which data is to be transferred (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: source address

destructive read. (1) read operation that alters the data in the accessed location (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: nondestructive read

DET. (1) data element type (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 4) (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)

detailed design. (1) process of refining and expanding the preliminary design of a system or component to the extent that the design is sufficiently complete to be implemented (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) result of the process in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: low-level design, software development process
detailed design description. (1) document that describes the exact detailed configuration of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It identifies the input, output, control logic, algorithms, and data structure of each individual low-level component of the software product and is the primary product of the detailed design phase. Syn: detailed design specification

detailed design phase. (1) software development lifecycle phase during which the detailed design process takes place, using the software system design and software architecture from the previous phase (architectural design) to produce the detailed logic for each unit such that it is ready for coding (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

detailed design review. (1) milestone review to determine the acceptability of the detailed software design (as depicted in the detailed design description) to satisfy the requirements of the software requirements document (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

detailed function point analysis. (1) the most accurate analysis to determine the size of an application or a project in which all the specifications needed are known in detail (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Note: Transactions have been specified up to the level of referenced logical files and data element types, and logical files have been specified up to the level of record types and data element types. As a result, the complexity of each function recognized can be established. See also: high level function point analysis, indicative function point analysis

Determine Budget. (1) the process of aggregating the estimated costs of individual activities or work packages to establish an authorized cost baseline (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

deterministic system. (1) system which, given a particular set of inputs and starting state, will always produce the same set of outputs and final state (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.28)

develop project charter. (1) the process of developing a document that formally authorizes the existence of a project and provides the project manager with the authority to apply organizational resources to project activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

develop project management plan. (1) the process of defining, preparing, and coordinating all subsidiary plans and integrating them into an integrated project management plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

develop schedule. (1) the process of analyzing activity sequences, durations, resource requirements, and schedule constraints to create the project schedule model for project execution and monitoring and controlling (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

developed source statement. (1) source statement that is newly created for, added to, or modified for a software product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

developer. (1) individual or organization that performs development activities (including requirements analysis, design, testing through acceptance) during the system or software life-cycle process (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.6)
development. (1) specification, construction, testing and delivery of a new application or of a discrete addition to an existing application (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) (2) activity of preparing information for users after it has been designed (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.9)

development approach. (1) the method used to create and evolve the product, service, or result during the project life cycle, such as predictive, iterative, incremental, agile, or a hybrid method (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: life cycle model

development branch. (1) branch where active product development takes place (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: A product build from the development branch will have the latest features, but will also likely be immature and unstable.

development environment. (1) hardware, software, platform and tools for designers and developers of computer solutions (ISO/IEC/IEEE 24765c:2014)

development plan. (1) plan for guiding, implementing, and controlling the design and development of one or more products or services (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: project plan

development project. (1) a project in which a completely new application is realized (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) project to develop and deliver the first release of a software application (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.18) Note: It entails the specification, construction, testing, and delivery of a new application. During actualization, this project can be split up into a number of sub-projects. If these are carried out more or less in parallel, each being responsible for effectuating a certain sub-system of the total application, then each sub-project can be considered as an individual development project, if the sub-system itself is an application. Re-building an existing application, otherwise known as re-engineering, is considered as development.

development project function point count (DFP). (1) a count that measures the functionality provided to the end users with the first installation of the software, developed when the project is complete (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) (2) activity of applying ISO/IEC 20926:2009 to measure the functional size of a development project (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.20)

development project functional size. (1) measure of the functionality provided to the users with the first release of the software, as measured by the development project function point count (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.19) Note: The functional size of a development project can include the size of conversion functionality.
development testing. (1) formal or informal testing conducted during the development of a system or component, usually in the development environment by the developer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) testing conducted to establish whether a new software product or software-based system (or components of it) satisfies its criteria (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The criteria will vary based on the level of test being performed. See also: acceptance testing, operational testing, qualification testing
development tool. (1) hardware and software for developing or modifying applications (ISO/IEC/IEEE 24765c:2014)
developmental baseline. (1) specifications that are in effect at a given time for a system under development (ISO/IEC 2382:2015 Information technology -- Vocabulary)
developmental configuration. (1) in configuration management, the software and associated technical documentation that define the evolving configuration of a computer software configuration item during development (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The developmental configuration is under the developer's control, and therefore is not called a baseline. See also: allocated baseline, functional baseline, product baseline
deviation. (1) departure from a specified requirement (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) written authorization, granted prior to the manufacture of an item, to depart from a particular performance or design requirement for a specific number of units or a specific period of time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
device. (1) mechanism or piece of equipment designed to serve a purpose or perform a function (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: platform
device interface task. (1) concurrent task that hides the characteristics of and interfaces to an external I/O device (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
DevOps. (1) set of principles and practices which enable better communication and collaboration between relevant stakeholders for the purpose of specifying, developing, and operating software and systems products and services, and continuous improvements in all aspects of the life cycle (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)
DFD. (1) data flow diagram (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
diagnostic. (1) pertaining to the detection and isolation of faults or failures (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
diagnostic manual. (1) document that presents the information necessary to execute diagnostic procedures for a system or component, identify malfunctions, and remedy those malfunctions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Typically described are the diagnostic features of the system or component and the diagnostic tools available for its support. See also: installation manual, operator manual, programmer manual, support manual, user manual
diagonal microinstruction. (1) microinstruction capable of specifying a limited number of simultaneous operations needed to carry out a machine language instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Diagonal microinstructions fall, in size and functionality, between horizontal microinstructions and vertical microinstructions. The designation 'diagonal' refers to this compromise rather than to any physical characteristic of
the microinstruction. See also: horizontal microinstruction, vertical microinstruction

diagram. (1) logically coherent fragment of a design view, using selected graphical icons and conventions for visual representation from an associated design language, to be used for representing selected design elements of interest for a system under design from a single viewpoint (diagram type) (2) instantiation of the formal diagram structure that consists only of semantically and syntactically valid IDEF0 graphical statements (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.39) Note: Each diagram is a single unit of an IDEF0 model that presents the top-level function that is the subject of the model (the A-0 context diagram), presents the context of the subject function (other context diagrams), or presents the details of a box (decomposition diagrams). See also: design subject


diagram number. (1) that part of a diagram reference that corresponds to a diagram's parent function's node number (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.43) Note: The diagram number refers to the diagram that details or decomposes the function designated by the same node number.

diagram page. (1) model page that contains a context diagram or a decomposition diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.44)

diagram reference. (1) expression that unambiguously identifies a diagram and specifies the diagram's position in a specific model hierarchy (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.45) Note: A diagram reference is composed of a model name, abbreviation and a diagram number.

diagram title. (1) verb or verb phrase that describes the overall function presented by a diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.46) Note: The diagram title of a child diagram is the box name of its parent box.

diagramming techniques. (1) approaches to presenting information with logical linkages that aid in understanding (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

dialog. (1) conversation between the user and the application needed to execute a transaction (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) interaction between a user and an interactive system as a sequence of user actions (inputs) and system responses (outputs) in order to achieve a goal (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.4) Syn: dialogue


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.

differential cash flow. (1) representation of the difference between cash flows of two alternatives or proposals (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: often performed using internal rate of return (IRR) as the basis of comparison See also: incremental analysis

digit numeric character. (1) character that represents a nonnegative integer (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: numeric character
digital. (1) pertaining to data that consists of digits as well as to processes and functional units that use the data (ISO/IEC 2382:2015 Information technology -- Vocabulary)
digital computer. (1) computer that is controlled by internally stored programs and that is capable of using common storage for all or part of a program and also for all or part of the data necessary for the execution of the programs; executing user-written or user-designated programs; performing user-designated manipulation of digitally represented discrete data, including arithmetic operations and logic operations; and executing programs that modify themselves during their execution (ISO/IEC 2382:2015 Information technology -- Vocabulary)
digital information content asset. (1) digital asset with information content (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.18) Note: Often licensed, but not considered to be software. IT asset management can include management of these assets for license compliance, but excludes management of the content.
digital signal processing (DSP). (1) modification of an information signal represented by a sequence of digits or symbols to affect the representation of discrete time, discrete frequency, or other attributes (ISO/IEC/IEEE 24765d:2015)

dimension. (1) distinct components that a multidimensional construct encompasses (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.6)

DIP. (1) dual inline package (ISO/IEC/IEEE 24765c:2014)
direct address. (1) address that identifies the storage location of an operand (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: one-level address See also: immediate data, indirect address, n-level address, direct instruction
direct and manage project work. (1) the process of leading and performing the work defined in the project management plan and implementing approved changes to achieve the project's objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
direct instruction. (1) computer instruction that contains the direct addresses of its operands (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: immediate instruction, indirect instruction, absolute
instruction, effective instruction

direct labor. (1) personnel efforts that are directly related to the units of production. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: indirect labor
direct measure. (1) measure of an attribute that does not depend upon a measure of any other attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
direct memory access (DMA). (1) technique in which a peripheral takes direct control of a central processing unit's memory bus to transfer data to or from memory. (ISO/IEC/IEEE 24765d:2015)
direct memory access controller (DMAC). (1) functional unit that performs direct memory access (ISO/IEC/IEEE 24765d:2015)
direct metric. (1) a metric that does not depend upon a measure of any other attribute. (IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.4)
direct staff-hour. (1) amount of effort directly expended in creating a specific output product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
directed graph. (1) a graph (sense 2) in which direction is implied in the internode connections (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: digraph See also: undirected graph
directed system of systems. (1) system of systems (SoS) created and managed to fulfill specific purposes and the constituent systems are subordinated to the SoS. (ISO/IEC/IEEE 21841:2019 Systems and software engineering--Taxonomy of systems of systems, 3.2.3) Note: Component systems maintain an ability to operate independently; however, their normal operational mode is subordinated to the central managed purpose. Syn: directed SoS
directory. (1) list of data items and information about those data items (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
disassemble. (1) to translate an assembled computer program from its machine language version into a form that resembles, but is not necessarily identical to, the original assembly language program. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assemble
disassembler. (1) software tool that disassembles computer programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
disaster recovery. (1) in computer system operations, the return to normal operation after a hardware or software failure (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) ability of the information and communications technology elements of an organization to support its critical business functions to an acceptable level within a predetermined period following a disaster (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.11)
disclaimer. (1) notice that renounces or repudiates a legal claim or right (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
discounted payback period. (1) time it will take to recover a project's initial investment including interest (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: An indication of exposure to risk. If a project is canceled before it reaches its payback period, the organization will have lost money.
discrete. (1) pertaining to data that consist of distinct elements, such as characters, or to physical quantities having a finite number of distinctly recognizable values, as well as to processes and functional units that use those data (ISO/IEC 2382:2015 Information technology -- Vocabulary)

discrete data. (1) data that arrives at specific time intervals (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

discrete effort. (1) work effort that is separate, distinct, and related to the completion of specific work breakdown structure components and deliverables, and that can be directly planned and measured (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) an activity that can be planned and measured and that yields a specific output (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: One of three earned value management types of activities used to measure work performance See also: apportioned effort

discrete type. (1) data type whose members can assume any of a set of distinct values (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A discrete type can be an enumeration type or an integer type.

discretionary dependency. (1) a relationship that is established based on knowledge of best practices within a particular application area or an aspect of the project where a specific sequence is desired. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: preferential logic, preferred logic, soft logic

discrimination (threshold). (1) largest change in a stimulus that produces no detectable change in the response of a measuring instrument, the change in the stimulus taking place slowly and monotonically (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.4) Note: The discrimination threshold can depend on, for example, noise (internal or external) or friction. It can also depend on the value of the stimulus.

discriminator. (1) property of a superclass, associated with a cluster of that superclass, whose value identifies to which subclass a specific instance belongs (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.51) (2) attribute in the generic entity (or a generic ancestor entity) of a category cluster whose values indicate which category entity in the category cluster contains a specific instance of the generic entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.51) Note: Since the value of the discriminator (when a discriminator has been declared) is equivalent to the identity of the subclass to which the instance belongs, there is no requirement for a discriminator in identity-style modeling. Syn: category discriminator

disk. (1) data medium originally consisting of a flat circular plate that is rotated in order to read or write data on one or both sides (ISO/IEC 2382:2015 Information technology -- Vocabulary)

display. (1) information presented on a screen or in a window of a screen (ISO/IEC/IEEE 24765g:2018)

disposal. (1) removal or archiving, but not deletion, of an artifact so it can be made available for traceability and auditability (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)

disposed system. (1) system that has been transformed (i.e. state change) by applying the disposal process (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.3) Note: A
systems approach considers the total system and the total lifecycle of the system. This includes all aspects of the system and the system throughout its life until the day users depose of the system and the external enterprises complete the handling of the disposed system products.

disposition. (1) range of processes associated with implementing retention, destruction or transfer decisions which are documented in disposition or other instruments (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.12)

distributed computing. (1) spreading of computation and data across a number of computers connected by a network (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
distributed processing. (1) information processing in which discrete components can be located in different places, and where communication between components can suffer delay or can fail (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.2)
distribution transparency. (1) property of hiding from the user some specific aspects of the system's complexity needed to support distribution (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.1.1)
distributional shift. (1) in machine learning, the distance between the training data distribution and the desired data distribution (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.29) Note: The effect of distributional shift often increases as the users interaction with the system (and so the desired data distribution) changes over time. Syn: dataset shift
disturbance. (1) operational fault or event or anything that could change the state of the system (ISO/IEC 25045:2010 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation module for recoverability, 4.2) Note: Disturbances are limited to external faults or events, rather than introduced internal faults that would require modifying the application or OS code.


DITA. (1) Darwin Information Typing Architecture (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 5)
diversity. (1) in fault tolerance, realization of the same function by different means (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software diversity
dividing action. (1) action which enables two or more chains (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.4)


DLC. (1) Data-Life-Cycle (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 5)

DMA. (1) direct memory access (ISO/IEC/IEEE 24765d:2015)

DMAC. (1) direct memory access controller (ISO/IEC/IEEE 24765d:2015)

DMS. (1) document management system (ISO/IEC/IEEE 24765h:2019)


DNS. (1) Domain Name Service (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

do nothing alternative. (1) in a decision analysis, the alternative of not investing in any of the proposed alternatives (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: doesn't really mean doing nothing at all. Instead, it means putting the money into readily available investments that give a predetermined rate of return (bonds, interest bearing accounts, put into a more profitable part of the organization)

document. (1) uniquely identified unit of information for human use (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 3.1.10) (2) to create a document as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to add comments to a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) separately identified piece of documentation which could be part of a documentation set (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.15) (5) medium, and the information recorded on it, that generally has permanence and can be read by a person or a machine (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (6) information and the medium on which it is contained (ISO/IEC 29110-4-3:2018 Systems and software engineering-Lifecycle profiles for very small entities (VSEs)-Part 4-3: Service delivery-Profile specification, 3.8) (7) fixed and structured amount of information intended for human perception that can be managed and interchanged as a unit between users and systems (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.10) Note: Documents include both paper and electronic documents. The documentation can be in any form or type of medium. Documents, except for records, state the intent to be achieved. A set of documents, for example specifications and records), is frequently called "documentation". See also: documentation

document control. (1) application of configuration management to the control of documents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

document management system (DMS). (1) system that supports the storage, retrieval, versioning, and manipulation of whole documents, images, and other media (ISO/IEC/IEEE 24765:2019) See also: content management system

document set. (1) collection of documentation that has been segmented into separately identified volumes or files for ease of distribution or use (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.17) (2) collection of documentation that has been segmented into separately identified volumes or products for ease of distribution or use (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.11)

document type definition (DTD). (1) template for the structure, content, and semantics of XML documents (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and
service management documentation, 4.13)

documentation. (1) collection of documents related to a given subject (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.10) (2) written or pictorial information describing, defining, specifying, reporting, or certifying activities, requirements, procedures, or results (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.14) (3) process of generating or revising a document (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (4) information that explains how to use a product (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.11) (5) management of documents, including identification, acquisition, processing, storage, and dissemination (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (6) information that explains how to use hardware, devices, applications, or services (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.10) Note: can be provided as separate documentation or as embedded documentation or both Syn: information for users See also: document

documentation reviews. (1) the process of gathering a corpus of information and reviewing it to determine accuracy and completeness (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: documentation review

documentation tree. (1) diagram that depicts all of the documents for a given system and shows their relationships to one another (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: specification tree

documented information. (1) information required to be controlled and maintained by an organization and the medium on which it is contained (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.19) Note: Documented information can be in any format and media and from any source. Documented information can refer to information created in order for the organization to operate (documentation), or evidence of results achieved (e.g. records, key performance indicators).


domain. (1) distinct scope, within which common characteristics are exhibited, common rules observed, and over which a distribution transparency is preserved (ISO/IEC 19500-2:2012 Information technology -- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.5) (2) problem space (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (3) area of knowledge or activity characterized by a set of concepts and terminology understood by practitioners in that area (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (4) distinct scope, within which common and variable characteristics are exhibited, common rules and binding mechanisms are observed, and over which a distribution transparency is preserved (ISO/IEC 26560:2019 Software and systems engineering -- Tools and methods for product line product management, 3.3) (5) aspect or area of knowledge or activity characterized by a set of concepts and terminology understood by practitioners in that area (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)
domain analysis. (1) analysis of systems within a domain to discover commonalities and differences among them (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (2) process by which information used in developing software systems is identified, captured, and organized so that it can be reused to create new systems, within a domain (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (3) result of the domain analysis process (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)

domain architecture. (1) generic, organizational structure or design for software systems in a domain (IEEE 1517-2010 IEEE Standard for Information Technology--System and software life cycle processes--Reuse processes, 3) (2) common architecture for a product line that can embrace variability of member products (ISO/IEC 26552:2019 Software and systems engineering--Tools and methods for product line architecture design, 3.5) (3) core architecture that captures the high-level design of a software and systems product line including the architectural structure and texture (e.g. common rules and constraints) that constrains all member products within a software and systems product line (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.10)

Note: The domain architecture contains the designs that are intended to satisfy requirements specified in the domain model. The domain architecture documents design, whereas the domain model documents requirements. A domain architecture: 1) can be adapted to create designs for software systems within a domain, and 2) provides a framework for configuring assets within individual software systems. The term "architecture" has been deliberately redefined to more properly convey its meaning in the software reuse context.

domain asset. (1) output of domain engineering life cycle processes that can be reused in producing products during application engineering (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.11) (2) output of a subprocess of domain engineering that is reused for producing two or more products in a product line (INCOSE Systems Engineering Handbook, 4th ed.) Note: Domain assets are not physical products available off-the-shelf and ready for commissioning. Domain assets have their own life cycles. Syn: core asset, domain artifact

domain assets in requirements. (1) reusable artifacts produced during domain requirements engineering (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.10)

domain component. (1) reusable component among member products within a product line (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.10)

domain engineering. (1) reuse-based approach to defining the scope (i.e., domain definition), specifying the structure (i.e., domain architecture), and building the assets for a class of systems, subsystems, or applications (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (2) life cycle consisting of a set of processes for specifying and managing the commonality and variability of a product line (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.12) Note: For example, "assets" such as requirements, designs, software code, documentation. Domain engineering can include the following activities: domain definition, domain analysis, developing the domain architecture, and domain implementation.

domain engineering process. (1) processes for domain asset development (ISO/IEC 26555:2015 Software and
domain interface. (1) reusable interface among the components of a member product (3.14) within a product line (ISO/IEC 26553:2018 Information technology - Software and systems engineering - Tools and methods for product line realization, 3.11)

domain layer. (1) highest level of abstraction for the test item (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering - Software testing - Part 5: Keyword-driven testing, 4.1)

domain model. (1) product of domain analysis that provides a representation of the requirements of the domain (ISO/IEC/IEEE 24765-2021) Note: The domain model identifies and describes the structure of data, flow of information, functions, constraints, and controls within the domain that are included in software systems in the domain. The domain model describes the commonalities and variabilities among requirements for software systems in the domain.

domain realization. (1) one of the domain engineering processes that include detailed design and implementation (ISO/IEC 26553:2018 Information technology - Software and systems engineering - Tools and methods for product line realization, 3.12)

domain requirements analysis. (1) subprocess that models domain requirements so as to analyze and scrutinize commonality/variability of a product line in requirements (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.12)

domain requirements elicitation. (1) subprocess that identifies initial requirements from domain stakeholders for a product line (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.11)

domain requirements management. (1) subprocess that manages traceability and changes with respect to domain requirements and their relevant domain/application artifacts (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.15)

domain requirements specification. (1) subprocess that documents domain requirements for a product line based on domain analysis results (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.13)

domain requirements verification and validation. (1) subprocess that confirms that domain requirements are correct, consistent, and complete (ISO/IEC 26551:2016 Software and systems engineering -- Tools and methods for product line requirements engineering, 3.14)

domain scoping. (1) subprocess for identifying and bounding the functional domains that are important to an envisioned product line and provide sufficient reuse potential to justify the product line creation (ISO/IEC 26550:2015 Software and systems engineering -- Reference model for product line engineering and management, 3.13) Note: Identifies and bounds the functional domains that are important to an envisioned product line and provide sufficient reuse potential to justify the product line creation.

domain test asset. (1) domain test artifacts that will be reused in application testing (ISO/IEC 26554:2018 Information technology -- Software and systems engineering - Tools and methods for product line testing, 3.6) Note: Domain test assets, e.g., test plans, artifacts of regression testing, can be reused in domain testing.

domain test requirement. (1) specific element of a domain artifact that should be covered by domain testing


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
Note: Domain test requirements cover functional and non-functional commonality and variability, and they include both their normal and error conditions.

domain testing. (1) domain engineering phase whose role is to test domain artefacts (ISO/IEC 26554:2018 Information technology--Software and systems engineering-Tools and methods for product line testing, 3.5) Note: testing of artifacts related to a product line (domain) rather than to an individual product

domain variability model. (1) explicit definition of product line variability (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.4)

domain-based requirement. (1) requirement originated from its application domain (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.5)

dominance. (1) decision technique that looks for an alternative that is at least as good in every attribute and better in at least one attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: lexicography, satisficing
done. (1) regarded by the agile team as complete and ready to use (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.5)
dot notation. (1) technique for naming that joins the name of a parent class to the name of a dependent class with the period character (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.47)
double data rate (DDR) SDRAM. (1) synchronous dynamic random access memory unit with higher access speed and bandwidth, because it transfers two consecutive words in one internal clock cycle (ISO/IEC/IEEE 24765c:2014)
down. (1) pertaining to a system or component that is not operational or has been taken out of service (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: up, busy, crash, idle
down time. (1) period of time during which a system or component is not operational or has been taken out of service (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: up time, busy time, idle time, mean time to repair, set-up time
downgrade right. (1) right granted to receive, install, or use an installation of a previous version of software than the currently granted entitlement (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.7)
download. (1) to transfer programs or data from a computer to a connected computer with fewer resources (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: typically, from a server to a personal computer
downward compatible. (1) pertaining to hardware or software that is compatible with an earlier or less complex version of itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: upward compatible
downward compression. (1) in software design, a form of demodularization in which a superordinate module is copied into the body of a subordinate module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: lateral compression, upward compression


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.

DR. (1) decision review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

DRAM. (1) dynamic random access memory (ISO/IEC/IEEE 24765c:2014)

drift. (1) changes to machine learning model behavior that occur over time (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 6.1.30) Note: These changes typically make predictions less accurate and can require the model to be re-trained with new data. Syn: degradation, staleness

driver. (1) software module that invokes and, perhaps, controls and monitors the execution of one or more other software modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) computer program that controls a peripheral device and, sometimes, reformats data for transfer to and from the device (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: test driver

DRP. (1) disaster recovery plan (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)


DSL. (1) definitive software library (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.11)


DT. (1) development test (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

DT&E. (1) developmental test and evaluation (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

DTC. (1) data transfer controller (ISO/IEC/IEEE 24765d:2015)


dual boot. (1) having more than one boot mode, to allow running two different operating systems on the same computer (ISO/IEC/IEEE 24765c:2015) See also: single boot

dual inline package (DIP). (1) microcircuit unit with connectors (pins) arranged in two rows (ISO/IEC/IEEE 24765c:2014)

dumb terminal. (1) user terminal that has no independent data processing capability (ISO/IEC 2382:2015 Information technology -- Vocabulary)
technology -- Vocabulary) Syn: nonprogrammable terminal
dump. (1) display of some aspect of a computer program’s execution state, usually the contents of internal storage or
registers (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) display of the contents of a file
or device (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to copy the contents of internal
storage to an external medium (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) to produce
a display or copy as in (1), (2), or (3) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note:
Types include change dump, dynamic dump, memory dump, postmortem dump, selective dump, snapshot dump, static
dump.
duration (DU or DUR). (1) The total number of work periods required to complete an activity or work breakdown
structure component,
expressed in hours, days, or weeks. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth
Edition) See also: effort
Engineering, 2.2) (2) digital video disc (ISO/IEC/IEEE 24765c:2014)
dyadic selective construct. (1) if-then-else construct in which processing is specified for both outcomes of the
branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: monadic selective construct
dynamic. (1) pertaining to an event or process that occurs during computer program execution (ISO/IEC/IEEE
24765:2017 Systems and software engineering-Vocabulary) See also: static
dynamic analysis. (1) process of evaluating a system or component based on its behavior during execution
(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: static analysis demonstration,
testing
dynamic binding. (1) binding performed during the execution of a computer program (ISO/IEC/IEEE 24765:2017
Systems and software engineering-Vocabulary) See also: static binding
dynamic breakpoint. (1) breakpoint whose predefined initiation event is a runtime characteristic of the program,
such as the execution of any twenty source statements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary) See also: static breakpoint, code breakpoint, data breakpoint, epilog breakpoint, programmable breakpoint,
prolog breakpoint
dynamic buffering. (1) buffering technique in which the buffer allocated to a computer program varies during
program execution, based on current need (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
See also: simple buffering
dynamic bus sizing. (1) capability to adjust the size of a bus on request during operations (ISO/IEC/IEEE
24765d:2015) Note: used during direct memory access
dynamic dump. (1) dump that is produced during the execution of a computer program (ISO/IEC/IEEE 24765:2017
Systems and software engineering-Vocabulary) See also: static dump, change dump, memory dump, postmortem dump,
selective dump, snapshot dump
dynamic error. (1) error that is dependent on the time-varying nature of an input (ISO/IEC/IEEE 24765:2017 Systems
and software engineering-Vocabulary) See also: static error
**dynamic invocation.** (1) constructing and issuing a request whose signature is possibly not known until run-time (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.6)

**dynamic model.** (1) model that describes individual requests or patterns of requests among objects (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.53) See also: static model

**dynamic product.** (1) system or software product that is measurable during execution in a testing or an operational environment (ISO/IEC 25041: 2012 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation guide for developers, acquirers and independent evaluators, 4.2)

**dynamic program analysis.** (1) process of evaluating a software system or component based on its behavior during execution (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.5) Note: The software is compiled and run on a certain number of input data test cases. The physical response from the system is then examined and compared to expected results. Dynamic program analysis can be done manually or using an automated process.

**dynamic random access memory (DRAM).** (1) RAM with a frequent refresh process to retain data (ISO/IEC/IEEE 24765c:2014) Note: used with a circuit architecture in single stable state

**dynamic relocation.** (1) relocation of a computer program during its execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**dynamic resource allocation.** (1) computer resource allocation technique in which the resources assigned to a program vary during program execution, based on current need (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**dynamic restructuring.** (1) process of restructuring a database, data structure, computer program, or set of system components during program execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**dynamic schema.** (1) specification of the allowable state changes of one or more information objects, subject to the constraints of any invariant schemata (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 6.1.3) Note: Behavior in an information system can be modeled as transitions from one static schema to another, i.e., reclassification of instances from one type to another. In the information language, a state change involving a set of objects can be regarded as an interaction between those objects. Not all of the objects involved in the interaction need change state; some of the objects can be involved in a read-only manner.

**dynamic skeleton.** (1) interface-independent kind of skeleton, used by servers to handle requests whose signatures are possibly not known until run-time (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.7)

**dynamic storage allocation.** (1) storage allocation technique in which the storage assigned to a computer program varies during program execution, based on the current needs of the program and of other executing programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**dynamic testing.** (1) testing that requires the execution of the test item (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.9) (2) testing that requires the execution of


EAC. (1) estimate at completion (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

early finish date (EF). (1) in the critical path method, the earliest possible point in time on which the uncompleted portions of a schedule activity can finish, based on the schedule network logic, the data date, and any schedule constraints (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

early start date (ES). (1) in the critical path method, the earliest possible point in time on which the uncompleted portions of a schedule activity can start, based on the schedule network logic, the data date, and any schedule constraints (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

early-failure period. (1) period of time in the life cycle of a system or component during which hardware failures occur at a decreasing rate as problems are detected and repaired (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: burn-in period See also: constant-failure period, wearout-failure period, bathtub curve

earned value (EV). (1) the measure of work performed expressed in terms of the budget authorized for that work (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: budgeted cost of work performed (BCWP)


echo. (1) to return a transmitted signal to its source, often with a delay to indicate that the signal is a reflection rather than the original (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) returned signal (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

ECMS. (1) Enterprise Content Management System (ISO/IEC/IEEE 26531:2015 Systems and software engineering--Content management for product lifecycle, user and service management documentation, 5)

ECO. (1) engineering change order (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)

economic risk mitigation. (1) degree to which a product or system mitigates the potential risk to financial status, efficient operation, commercial property, reputation, or other resources in the intended contexts of use (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.4.1)

ECP. (1) engineering change proposal (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

ECR. (1) engineering change request (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

EDI. (1) electronic data interchange (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
edit. (1) to modify the form or format of computer code, data, or documentation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
EDRAP. (1) engineering data requirements agreement plan (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
EEPROM. (1) electric erasable programmable read only memory (ISO/IEC/IEEE 24765c:2014)
effective address. (1) address that results from performing any required indexing, indirect addressing, or other address modification on a specified address (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: If the specified address requires no modification, it is also the effective address. See also: generated address, indirect address, relative address
effective instruction. (1) computer instruction that results from performing any required indexing, indirect addressing, or other modification on the addresses in a specified computer instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: If the specified instruction requires no modification, it is also the effective instruction. See also: absolute instruction, direct instruction, immediate instruction, indirect instruction
effective interest rate. (1) interest rate that has been adjusted for more or less frequent compounding (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
efferent. (1) pertaining to a flow of data or control from a superordinate module to a subordinate module in a software system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: afferent
efficiency. (1) degree to which a system or component performs its designated functions with minimum consumption of resources (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) resources expended in relation to the accuracy and completeness with which users achieve goals (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.3) (3) relationship between the result achieved and the resources used (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.17) Note: Time-on-task and Completion Rate/Mean Time-On-Task (defect rates vs. time to achieve task) are measures of efficiency. Efficiency in the context of usability is related to productivity, rather than to its meaning in the context of software efficiency. Efficiency is the degree to which an information system efficiently uses the technical infrastructure and thus becomes usable for the customer. The most important underlying topic here is the capacity of the platform in relation to the demand. See also: execution efficiency, storage efficiency
effort. (1) the number of labor units required to complete a schedule activity or work breakdown structure component.
often expressed in hours, days or weeks. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* See also: duration
goless programming. *(1)* software development technique based on the concept of team, rather than individual, responsibility for program development *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: Its purpose is to prevent individual programmers from identifying so closely with their work that objective evaluation is impaired.
EIF. *(1)* external interface file *(ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 4)*
electric erasable programmable read only memory (EEEPROM). *(1)* type of programmable ROM in which the memory can be erased using electrical current and rewritten *(ISO/IEC/IEEE 24765c:2014)* See also: flash memory
electronic design automation (EDA). *(1)* software-driven design and development of electronic components such as microcomputer units and circuit boards *(ISO/IEC/IEEE 24765d:2015)*
electronic mail (Email). *(1)* correspondence in the form of messages transmitted over a computer network *(ISO/IEC 2382:2015 Information technology -- Vocabulary)* Syn: e-mail
electronic publishing. *(1)* production of typeset-quality documents including text, graphics, and pictures with the assistance of a computer *(ISO/IEC 2382:2015 Information technology -- Vocabulary)*
element type. *(1)* category or class of elements *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*
elementary process. *(1)* smallest unit of activity that is meaningful to the user *(ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.21)* *(2)* a function
when two criteria are satisfied: a) The function has an autonomous meaning to the user and fully executes one complete processing of information. In other words, it is self-contained; b) After the function has been executed, the application is in a consistent state (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Syn: EP
ELF. (1) external logical file (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)
ELSE-rule. (1) actions to be taken for all combinations of conditions not covered by the other rules in the table (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.5) Note: The use of the ELSE-rule facility is optional.
embedded computer system. (1) computer system that is part of a larger system and performs some of the requirements of that system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The hardware and software of an embedded system are usually minimized and optimized for specific functions. The embedded system includes at least one microcontroller, microprocessor or digital signal processor. The embedded system designed to optimize reliability, cost, size and power saving for applications. Syn: embedded system
embedded documentation. (1) information that is delivered as an integral part of a piece of software (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.14) See also: separate documentation
embedded help system. (1) information for users that is delivered as an integral part of a piece of software (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.14)
embedded middleware. (1) software that communicates between an embedded operating system and an embedded application or firmware (ISO/IEC/IEEE 24765e:2015)
embedded operating system. (1) operating system software for an embedded computer system (ISO/IEC/IEEE 24765d:2015)
embedded software. (1) software that is part of a larger system and performs some of the requirements of that system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
EMC. (1) electromagnetic compatibility (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
EMD. (1) engineering and manufacturing development (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
emergence. (1) principle that entities exhibit properties which are meaningful only when attributed to the whole, not to its parts (ISO/IEC/IEEE 21840:2019 Systems and software engineering--Guidelines for the utilization of ISO/IEC/IEEE 15288 in the context of system of systems (SoS), 3.1.3) Note: These properties cannot be reduced or decomposed back down to the those of any individual constituent system.
emergency maintenance. (1) unscheduled modification performed to temporarily keep a system operational pending corrective maintenance (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.3) Note: Emergency maintenance is a part of corrective maintenance.
EMI. (1) electromagnetic interference (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

emitter. (1) event source that can be connected to at most one consumer (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

emotional intelligence. (1) the capability to identify, assess, and manage the personal emotions of oneself and other people, as well as the collective emotions of groups of people (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

emulated user. (1) imitation of a user, with regard to the tasks he submits and his time behavior, realized by a technical system (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.6)

emulation. (1) model that accepts the same inputs and produces the same outputs as a given system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of developing or using a model (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) use of a data processing system to imitate another data processing system, so that the imitating system accepts the same data, executes the same programs, and achieves the same results as the imitated system (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: simulation

evaluator. (1) device, computer program, or system that accepts the same inputs and produces the same outputs as a given system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: often used for testing or debugging See also: simulator

enabled behavior. (1) behavior characterizing a set of objects which becomes possible as a result of establishing behavior (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.2.2) Syn: enabled behaviour

enabling (a transition). (1) state of a transition in a particular mode and net marking when the following conditions are met: the marking of each input place of the transition satisfies the demand imposed on it by its arc annotation evaluated for the particular transition mode; the demand is satisfied when the place's marking contains (at least) the multiset of tokens indicated by the evaluated arc annotation; the determination of transition modes guarantees that the transition condition is satisfied (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.8) Note: The determination of transition modes guarantees that the transition condition is satisfied.

enabling system. (1) system that supports a system-of-interest during its life cycle stages but does not necessarily contribute directly to its function during operation (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.20) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.18) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.19) Note: For example, when a system-of-interest enters the production stage, an enabling production system is required. Each enabling system has a life cycle of its own. See also: software development environment

enactment. (1) establishment of something by law, ruling, or other authoritative acts (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
Systems and software engineering-Vocabulary) (2) act of applying a methodology for some particular purpose, typically an endeavor (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.8)

encapsulation. (1) software development technique that consists of isolating a system function or a set of data and operations on those data within a module and providing precise specifications for the module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) concept that access to the names, meanings, and values of the responsibilities of a class is entirely separated from access to their realization (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.54) (3) idea that a module has an outside that is distinct from its inside, that it has an external interface and an internal implementation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data abstraction, information hiding

encoding. (1) definition of how the elements of a syntax are represented using an identified character set (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Details of representation of the various terminal symbols and data types in the syntax's grammar are provided.

ENCODING.1. (1) primary encoding defined within the CDIF family of standards (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The CDIF family of standards supports multiple transfer formats, each composed of a syntax and an encoding.

end item. (1) entity that is ready for use (ISO/IEC/IEEE 24765:2016)

detail of period convention. (1) representation of discrete cash-flow instances at the end of the period in which they occur (in contrast to showing them at the beginning) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The initial investment is shown at the end of period zero.

end user. (1) person who directly uses the system for its intended purpose (ISO/IEC/IEEE 24765e:2015) (2) the person or persons who will ultimately be using the system for its intended purpose (3) individual person who ultimately benefits from the outcomes of the system or software (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.7) (4) any person that communicates or interacts with the software at any time (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size measurement method, 3.5) (5) person or persons who will ultimately be using the system for its intended purpose (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.13) (6) individual person who ultimately benefits from the ready-to-use software product functionalities (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for Quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.7) Note: An end user will generally be defined in terms of a specific software component of a system. Syn: end-user See also: direct user, functional user, indirect user, operator, secondary user, user

endeavor. (1) IBD development effort aimed at the delivery of some product or service through the application of a methodology (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.5) Syn: endeavour

endeavor element. (1) simple component of an endeavor (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.7) Note: During the execution of an endeavor, developers create a number of endeavor elements, such as tasks, models, classes, documents.
endurance testing. (1) type of performance efficiency testing conducted to evaluate whether a test item can sustain a required load continuously for a specified period of time (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.10)


engineering change. (1) alteration in the configuration of a hardware/software configuration item or items, delivered, to be delivered, or under development, after formal establishment of their configuration identification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in configuration management, an alteration in the configuration of a configuration item or other designated item after formal establishment of its configuration identification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control, engineering change proposal, deviation, waiver

engineering change proposal (ECP). (1) in configuration management, a proposed engineering change and the documentation by which the change is described and suggested (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control

engineering interface reference. (1) identifier, in the context of an engineering interface reference management domain, for an engineering object interface that is available for distributed binding (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.16) Note: An engineering interface reference is necessary to establish distributed bindings, and is distinct from the binding endpoint identifiers used by a basic engineering object for the purposes of interaction.


engineering interface reference management policy. (1) set of permissions and prohibitions that govern the federation of engineering interface reference management domains (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.18)

engineering viewpoint. (1) viewpoint on an ODP system and its environment that focuses on the mechanisms and functions required to support distributed interaction between objects in the system (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.1.1.4)

ehancement. (1) the activities performed for an application that change the specifications of the application (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: These activities usually also change the functional size as a result (e.g. change requests). See also: change

enhancement project. (1) project to develop and deliver adaptive maintenance (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.22) (2) a project in which enhancements are made to an existing application (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)
Note: In an enhancement project, functionality can be added to, changed in, or deleted from an existing application. An enhancement project can also develop and deliver corrective and perfective maintenance, but these do not contribute to the enhancement project functional size.

**enhancement project function point analysis.** (1) an analysis that measures a project that realizes modifications to an existing application, which is the addition, change, or deletion of functions (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)

**enhancement project function point count (EFP).** (1) activity of applying ISO/IEC 20926:2009 to measure the functional size of an enhancement project (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.24)

**ensure.** (1) to make certain that things occur or events take place (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: Insure is used only for insurance matters. See also: assure

**Ent.** (1) [software] entitlement schema (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.2)

**Ent creator.** (1) entity that initially creates an Ent (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.10) Note: This entity can be part of the organization that created or published the software to which the Ent relates, in which case the Ent creator and software creator will be the same. The Ent creator can also be a separate organization which holds the licensing rights or even a third-party organization unrelated to the software creator (such as in the case where Ents are created for legacy software by a consultant or tool developer).

Syn: entitlement schema creator

**enterprise.** (1) bold or complex endeavor (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.9) (2) purposeful combination of interdependent resources that interact with each other to achieve business and operational goals (INCOSE Systems Engineering Handbook, 4th ed.) Note: One or more organizations can participate in an enterprise. In case of multi-organization enterprises, each of the organizations brings various resources forward for use in the enterprise and they participate to the extent that they benefit from their involvement. The purpose of the enterprise is to address some challenges that these participating organizations cannot readily address on their own. Within a single organization, an enterprise may refer to a subset of the organization which is typically addressing particularly challenging or complex issues, often over a defined duration, and may undertake this with certain relaxations, tightening or otherwise authorized modifications of its standard processes and practices.

**enterprise environmental factors.** (1) conditions, not under the immediate control of the team, that influence, constrain or direct the project, program or portfolio (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**enterprise viewpoint.** (1) viewpoint on an ODP system and its environment that focuses on the purpose, scope, and policies for that system (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.1.1.1)

**entitlement schema.** (1) information structure containing a digital encapsulation of a licensing transaction and its
associated entitlement information (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.11) Note: A single transaction does not necessarily encapsulate a full (or effective) entitlement. An effective entitlement can be determined by an analysis of multiple licensing transactions, of a full license and then of upgrades and/or maintenance transactions assessed together with it. Syn: software entitlement schema, Ent

entity. (1) a fundamental thing of relevance to the user, about which information is kept (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) (2) in computer programming, any item that can be named or denoted in a program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) the representation of a concept, or meaning, in the minds of the people of the enterprise (4) object (i.e., thing, event or concept) that occurs in a model (i.e., transfer) (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (5) object that is to be characterized by measuring its attributes (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.9) (6) representation of a set of real or abstract things that are recognized as the same type because they share the same characteristics and can participate in the same relationships (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.55) (7) concrete or abstract thing of interest (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 6.1) (8) object to be modeled (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.3) (9) logical component of the data store, representing fundamental things of relevance to the user, and about which persistent information is stored (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FiSMA 1.1 functional size measurement method, A.8) (10) registered organization, group within a registered organization, or a project within an organization (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.26) Note: While in general the word entity can be used to refer to anything, in the context of modeling it is reserved to refer to things in the universe of discourse being modeled.

text entity component. (1) CORBA component with persistent state, identity which is architecturally visible to clients through a primary key, and behavior, which can be transactional (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

text entity dependent. (1) #9001;entity#9002; not meaningful or not significant to the business in and of itself without the presence of other entities, such that an occurrence of entity X must be linked to an occurrence of entity Y, and the deletion of an occurrence of entity Y results in the deletion of all related occurrences of entity X (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.25) Syn: entity-dependent

text entity independent. (1) #9001;entity#9002; meaningful or significant to the business in and of itself without the presence of other entities (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.26) Syn: entity-independent

text entity instance. (1) one of a set of real or abstract things represented by an entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.56) Note: Each instance of an entity can be specifically identified by the value of the attribute(s) participating in its primary key. [key style]
text entity-relationship (E-R) diagram. (1) a diagram that depicts a set of real-world entities and the logical
relationships among them. Syn: entity-relationship map See also: data structure diagram

**entry (n-type).** (1) data movement that moves a data group from a functional user across the boundary into the functional process where it is required. (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.8) Note: an entry is considered to account for certain associated data manipulations (e.g., validation of the entered data) Syn: entry type

**entry criteria.** (1) states of being that must be present before an effort can begin successfully. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) artifacts and other review or audit elements that must be completed before the review or audit can be conducted. (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1) See also: exit criteria

**entry field.** (1) area on a screen or in a window in which a user enters data. (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.19)

**entry point.** (1) point in a software module at which execution of the module can begin. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) point in a test item at which execution of the test item can begin. (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.18) Note: An entry point is an executable statement within a test item that can be selected by an external process as the starting point for one or more paths through the test item. It is most commonly the first executable statement within the test item. Syn: entrance, entry. See also: exit, reentry point

**entry profile.** (1) profile targeted at start-up Very Small Entities (i.e., VSEs that started their operation fewer than three years ago) or at VSEs working on small projects (e.g., project size of less than six person-months). (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.27)

**enumeration type.** (1) discrete data type whose members can assume values that are explicitly defined by the programmer. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: character type, integer type, logical type, real type

**environment.** (1) [system] context determining the setting and circumstances of all influences upon a system. (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.21) (ISO/IEC/IEEE 42010:2011 Systems and software engineering--Architecture description, 3.8) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.19) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.20) (2) configuration of hardware and software in which the software operates. (ISO/IEC/IEEE 24765a:2011) (3) of an object, the part of the model which is not part of that object. (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.2) (4) context determining the setting and circumstances of influences upon an architecture entity or upon which the architecture entity can have an influence. (ISO/IEC/IEEE 42030:2019 Software, systems, and enterprise--Architecture evaluation framework, 3.7) (5) concept space, i.e., an area in which a concept has an agreed-to meaning and one or more agreed-to names that are used for the concept. (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.57) (6) surroundings (natural or human)#8208;made in which the system of interest is utilized and supported or in which the system is being developed,
produced, and retired (INCOSE Systems Engineering Handbook, 4th ed.) Note: The environment of a system includes developmental, technological, business, operational, organizational, political, economic, legal, regulatory, ecological and social influences.

evironment contract. (1) contract between an object and its environment, including Quality of Service constraints, usage and management constraints (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.3)

environmental risk mitigation. (1) degree to which a product or system mitigates the potential risk to property or the environment in the intended contexts of use (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.4.3)


EP. (1) elementary process (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)

epic. (1) a high-level or complex user story to be refined into more detailed user stories (Software Extension to the PMBOK(R) Guide Fifth Edition) (2) major collection of related feature sets broken down into individual features or user stories and implemented in parts over a longer period of time (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.6)

epilog breakpoint. (1) breakpoint that is initiated upon exit from a given program or routine (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: postamble breakpoint See also: prolog breakpoint, code breakpoint, data breakpoint, dynamic breakpoint, programmable breakpoint, static breakpoint

epoch. (1) period of time for which an object displays a particular behavior (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 10.5)

EPROM. (1) erasable programmable read only memory (ISO/IEC/IEEE 24765c:2014)


equipment. (1) associated assemblies intended to achieve a defined final objective (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.12)

equivalence class. (1) range on a classification axis which has a rule to judge whether a target system is to be mapped to the range or not (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.8)

equivalence partition. (1) subset of the range of values of a variable, or set of variables, within a test item or at its interfaces, such that all the values in the partition can reasonably be expected to be treated similarly by the test item (i.e., they are considered "equivalent") (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.11)

equivalence partition coverage. (1) proportion of identified equivalence partitions of a test item that are covered
by a test set (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.12) Note: In many cases, the identification of equivalence partitions is subjective (especially in the sub-partitioning of "invalid" partitions), so a definitive count of the number of equivalence partitions in a test item could be impossible.

equivalence partitioning. (1) test design technique in which test cases are designed to exercise equivalence partitions by using one or more representative members of each partition (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.13)

equivalent faults. (1) two or more faults that result in the same failure mode (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

equivalent IDL. (1) client mappings; that is, mappings of the externally-visible component features for component declarations, or home features for home declarations (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: Implicitly defined by a component definition in IDL (interface definition language)

equivalent interface. (1) interface that manifests the component's or home's surface features to clients, allowing clients to navigate among the component's facets, and to connect to the component's ports, as defined by the component's or home's equivalent interface definition language (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)


erasable programming read only memory (EPROM). (1) type of programmable ROM which can be rewritten after erasing the existing data using ultraviolet (UV) rays (ISO/IEC/IEEE 24765c:2014) Note: The device can be rewritten many times.

ergonomics. (1) scientific discipline concerned with the understanding of the interactions among human and other elements of a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

errata. (1) severe service-disrupting bugs for which there is no known workaround (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Fixes for such bugs can often be introduced on a frozen branch.

error. (1) discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.5) See also: failure, defect

error guessing. (1) test design technique in which test cases are derived on the basis of the tester's knowledge of past failures, or general knowledge of failure modes (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.14)

error message. (1) a message that the application gives when incorrect data is entered or when another processing error occurs (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)
error model. (1) in software evaluation, a model used to estimate or predict the number of remaining faults, required test time, and similar characteristics of a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: error prediction model

error prediction. (1) quantitative statement about the expected number or nature of faults in a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: error model, error seeding

error processing. (1) process of detecting and responding to a program’s errors (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

error seeding. (1) process of intentionally adding known faults to those already in a computer program for the purpose of monitoring the rate of detection and removal, and estimating the number of faults remaining in the program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: bug seeding, fault seeding See also: indigenous error

error tolerance. (1) ability of a system or component to continue normal operation despite the presence of erroneous inputs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: fault tolerance, robustness


escaped. (1) preceding each occurrence of a pattern by the <EscapeCharacter>, if it is necessary to include a pattern in the text string that matches the <CloseText> delimiter (ISO/IEC 15475-3:2002 Information technology -- CDIF transfer format -- Part 3: Encoding ENCODING.1, 7.2.11)

escrow. (1) source code and documentation that is kept in the custody of a third party until specified contractual conditions have been fulfilled (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.20)


ESOH. (1) environment, safety, and occupational health (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

establish and maintain. (1) to formulate, document, and use [a policy or procedure] throughout an organization (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This phrase means more than a combination of its component terms; it includes documentation and usage. See also: maintain

established requirement. (1) requirement that the project has verified as satisfying project-specific criteria (such as clarity, suitability, and feasibility) and has validated to be an accurate representation of stakeholder needs, wants, and expectations (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: Established requirements are accepted by the project to form the basis of product development.

establishing behavior. (1) behavior by which a given contract is put in place between given objects (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.2.1) Syn: establishing behaviour

estimate. (1) quantitative assessment of the likely amount or outcome of a variable, such as project costs, resources, effort, or durations (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: usually preceded by a modifier (i.e., preliminary, conceptual, feasibility, order-of-magnitude, definitive) and including an
indication of accuracy (e. g., (+ or -) x percent). See also: budget, cost

estimate activity durations. (1) the process of estimating the number of work periods needed to complete individual activities with the estimated resources. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

estimate activity resources. (1) the process of estimating team resources and the type and quantities of material, equipment, and supplies necessary to perform project work. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

estimate at completion (EAC). (1) the expected total cost of completing all work expressed as the sum of the actual cost to date and the estimate to complete. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: earned value technique, estimate to complete

estimate costs. (1) the process of developing an approximation of the monetary resources needed to complete project activities. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

estimate to complete (ETC). (1) the expected cost to finish all the remaining project work. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: estimation

estimation. (1) process of developing a quantitative assessment of the likely amount or outcome. (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.5)


ETL. (1) extract, transform, load. (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)


evaluation. (1) systematic determination of the extent to which an entity meets its specified criteria. (ISO/IEC 25001:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Planning and management, 4.1) (2) action that assesses the value of something. (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.7) Note: Value can be considered in terms of usefulness, importance, preference, acceptability, etc.; the evaluated target can be, for example, a credit rating, a system state, a potential behavior.

evaluation activity. (1) assessment of systems or software product against targeted values of identified and applicable quality characteristics performed using applicable techniques or methods. (ISO/IEC 25001:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Planning and management, 4.1)

evaluation checklist. (1) list of questions, each of which is designed to check for conformity of a product, process or service to one or more provisions within a particular International Standard. (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.2)
evaluation coverage. (1) degree to which the evaluation covers the specified software product quality requirements (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.17)

evaluation group. (1) organization responsible for specifying the systems and software quality requirements as well as managing and implementing the quality evaluation activities through the provision of technology, tools, experiences, and management skills (ISO/IEC 25001:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Planning and management, 4.3) Note: Software quality requirements could be specified previously by the requestor of the evaluation, while the evaluation group would verify presence and value of the software quality requirements.

evaluation level. (1) rigor to be applied during the evaluation that defines the depth or thoroughness of the evaluation in terms of evaluation techniques to be applied and evaluation results to be achieved (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.18)

evaluation method. (1) procedure describing actions to be performed by the evaluator in order to obtain results for the specified measurement applied to the specified product components or on the product as a whole (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.8)

evaluation module. (1) package of evaluation technology for measuring software quality characteristics, subcharacteristics, or attributes (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.9) (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.20) Note: The package includes evaluation methods and techniques, input to be evaluated, data to be measured and collected, and supporting procedures and tools.

evaluation module (EVM). (1) microcomputer module used in application development, e.g., to benchmark software, prototype applications, and debug algorithms for computer systems (ISO/IEC/IEEE 24765d:2015)

evaluation procedure. (1) series of tasks and steps that, when completed, enable the evaluation team to determine if the product, process or service being evaluated is conformant to a particular standard (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.3)

evaluation records. (1) documented objective evidence of all activities performed and of all results achieved within the evaluation process (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.21)

evaluation report. (1) system follow-up report that describes how the system objectives have been met, identifies the remaining problems, and is intended to assist future development (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) document that presents evaluation results and other information relevant to an evaluation (ISO/IEC/IEEE 24765j:2021)

evaluation requester. (1) person or organization that requests an evaluation (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.22)
evaluation sponsor. (1) person or organization that requires the evaluation to be performed and provides financial or other resources to carry it out (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.4)

evaluation stringency. (1) degree required for the software product quality characteristics and sub-characteristics to fulfill the expected use criticality of the software product (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.24)

evaluation technology. (1) techniques, processes, tools, measures and relevant technical information used for evaluation (ISO/IEC 25001:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Planning and management, 4.3) Syn: technology used for evaluation

evaluation tool. (1) instrument that can be used during evaluation to collect data, to perform interpretation of data or to automate part of the evaluation (ISO/IEC/IEEE 24765:2021)

evaluator. (1) individual or organization that performs an evaluation (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.18)

(2) competent person engaged in the verification or validation of a system or software (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.7)

event. (1) occurrence of a particular set of circumstances (ISO/IEC/IEEE 24765:2021) (2) fact that an action has taken place (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.4) (3) external or internal stimulus used for synchronization purposes (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: The event can be certain or uncertain. The event can be a single occurrence or a series of occurrences, and can have several causes. [ISO Guide 73:2009, definition 3.6.1.3] The probability associated with the event can be estimated for a given period of time. An event can be an external interrupt, a timer expiration, an internal signal, or an internal message.

event history. (1) object representing significant actions (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.1.1.1)

event sequence analysis. (1) performance analysis of the sequence of tasks that must be executed to service a given external event (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

event sequence diagram. (1) diagram that identifies the sequence of tasks required to process an external event (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

event sink. (1) operation interface consuming announcements carrying notifications of typed events (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.16) (2) named connection point into which events of a specified type can be pushed (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) See also: consumer

event source. (1) operation interface originating announcements carrying notifications of typed events (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.16) (2) named connection point that emits events of a specified type to one or more interested event consumers, or to an event channel (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)
Broker Architecture (CORBA)--Part 3: Components, 4.1)

**event synchronization.** (1) control of task activation by means of signals (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Three types of event synchronization are possible: external interrupts, timer expiration, and internal signals from other tasks.

**event trace.** (1) time-ordered description of each external input and the time at which it occurred (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**event-sequencing logic.** (1) description of how a task responds to each of its message or event inputs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: in particular, what output is generated as a result of each input.


**examination.** (1) mechanism that is part of the assessment, which measures a candidate’s competence by one or more means such as written, oral, practical and observational, as defined in the certification scheme (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.10)

**exception.** (1) event that causes suspension of normal program execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) indication that an operation request was not performed successfully (ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3.8) Note: Types include addressing exception, data exception, operation exception, overflow exception, protection exception, and underflow exception.

**exception handling.** (1) programming language mechanism that passes error information by throwing and catching exceptions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**exclusive requirement.** (1) requirement of a normative document that must necessarily be fulfilled in order to comply with that document (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.5) Note: deprecated: mandatory requirement. [ISO/IEC Guide 2:2004]

**executable requirements specification.** (1) software requirement specification that is represented in an executable requirements language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**executable source statement.** (1) source statement that directs the actions of the computer at run time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**executable statement.** (1) statement which, when compiled, is translated into object code, which will be executed procedurally when the test item is running and can perform an action on program data (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.19)

**execute.** (1) to carry out an instruction, process, or computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) directing, managing, performing, and accomplishing the project work, providing the deliverables, and providing work performance information. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
executing process group. (1) those processes performed to complete the work defined in the project management plan to satisfy the project requirements (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

execution efficiency. (1) degree to which a system or component performs its designated functions with minimum consumption of time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: execution time, storage efficiency

execution time. (1) time which elapses between task submission and completion (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.7) (2) amount of elapsed time or processor time used in executing a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Processor time is usually less than elapsed time because the processor can be idle (for example, awaiting needed computer resources) or employed on other tasks during the execution of a program. See also: run time

execution trace. (1) record of the sequence of instructions executed during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: often takes the form of a list of code labels encountered as the program executes Syn: code trace, control flow trace See also: retrospective trace, subroutine trace, symbolic trace, variable trace

executor. (1) programming artifacts that supply the behavior of a component or a component home (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

existence constraint. (1) constraint stating that an instance of one entity cannot exist unless an instance of another related entity also exists (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.59) Note: [key style]

existence dependency. (1) constraint between two related entities indicating that no instance of one can exist without being related to an instance of the other (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.60) Note: The following association types represent existence dependencies: identifying relationships, categorization structures and mandatory nonidentifying relationships. [key style]

existing software. (1) software that is already developed and available; is usable either &quot;as is&quot; or with modifications; and which is provided by the supplier, acquirer, or a third party (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

exit. (1) point in a software module at which execution of the module can terminate (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) data movement that moves a data group from a functional process across the boundary to the functional user that requires it (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.9) Note: An exit is considered to account for certain associated data manipulations (e.g. formatting and routing associated with the data to be exited). Syn: exit type See also: entry point, return exit criteria. (1) states of being that must be present before an effort can end successfully (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) review or audit elements that must be assessed, completed, and
action items closed before successful completion of the technical review or audit can be declared (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1) See also: entry criteria

exit point. (1) last executable statement within a test item (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.20) Note: An exit point is a terminal point of a path through a test item, being an executable statement within the test item which either terminates the test item, or returns control to an external process. This is most commonly the last executable statement within the test item.

exit routine. (1) routine that receives control when a specified event, such as an error, occurs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

expandability. (1) degree of effort required to improve or modify software functions’ efficiency (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: extendability

expected results. (1) observable predicted behavior of the test item under specified conditions based on its specification or another source (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.15)

expected value. (1) estimated outcome that is as likely to be exceeded as not (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: the mean of the probability distribution, the point where the cumulative probability function equals 0.5 Syn: 50-50 estimate

expected value of perfect information. (1) in decision-tree analysis, the difference between the expected value of the decision tree and the value of the decision tree if all random outcomes were known in advance (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: helps the decision maker determine whether it is justifiable to invest in activities that would reduce uncertainties

expert judgment. (1) judgment provided based upon expertise in an application area, knowledge area, discipline, industry, etc. as appropriate for the activity being performed. Such expertise may be provided by any group or person with specialized education, knowledge, skill, experience, or training (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

expert system (ES). (1) computer system that provides for expertly solving problems in a given field or application area by drawing inferences from a knowledge base developed from human expertise (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: Some expert systems are able to improve their knowledge base and develop new inference rules based on their experience with previous problems.

explanatory report. (1) document attached to a product for providing complementary information in order to assist understanding and to avoid inappropriate usage of the product (ISO/IEC 29155-3:2015 Systems and software engineering--Information technology project performance benchmarking framework--Part 3: Guidance for reporting) Note: Examples of an explanatory report are data element definitions, data demographics, data source information which are attached to benchmarking repositories or benchmarks. Examples of the product are benchmarking repository, benchmark(s), or software tools to support benchmarking activities.

explicit knowledge. (1) knowledge that can be codified using symbols such as words, numbers, and pictures. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

exploitable. (1) characteristic of a vulnerability that can be activated in practice (ISO/IEC 23643:2020, Software and Systems Engineering Vocabulary)
Exploratory testing. (1) type of unscripted experience-based testing in which the tester spontaneously designs and executes tests based on the tester’s existing relevant knowledge, prior exploration of the test item (including the results of previous tests), and heuristic &quot;rules of thumb&quot; regarding common software behaviors and types of failure (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.9) Note: Exploratory testing hunts for hidden properties (including hidden behaviors) that, while quite possibly benign by themselves, could interfere with other properties of the software under test, and so constitute a risk that the software will fail.

Export process. (1) process of generating a transfer file from a source environment (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

Exporter. (1) agent of the export process (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

Extend. (1) in UML, a relationship from an extending use case to a base use case, specifying how the behavior defined for the extending use case can be optionally inserted into the behavior defined for the base use case (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

Extendability. (1) ease with which a system or component can be modified to increase its storage or functional capacity (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: extensibility See also: expandability, flexibility, maintainability


Extended entry table. (1) decision table where the conditions and actions are generally described but are incomplete (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.15) Note: The specifications are completed by the values specified in the rules

Extended process set. (1) set of processes specific to a maturity level higher than the basic maturity level that ensures the achievement of the relevant process profile (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.5)

Extensible markup language (XML). (1) license-free and platform-independent markup language that carries rules for generating text formats that contain structured data (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.15) (2) formal language used to specify the structure of XML documents (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.15) Note: specified in the XML Schema Part 1 Structures Recommendation

EXtensible Style Language Transformations (XSLT). (1) language for transforming XML documents into other document types, such as PDF or HTML (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.17) Syn: extensible style language transformations

Extension of a type. (1) set of entities that satisfy the type at any particular time (ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces,
5.3.4) **extensional set.** (1) set containing the currently existing instances of a class *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.61)* Note: The instances in the extensional set correspond to the database and data modeling notion of instance. Syn: current extent

**external.** (1) input information source or output information destination that is outside the scope of the project life cycle *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)* See also: invocation, iteration, mapping

**external attribute.** (1) measurable property of an entity which can only be derived with respect to how it relates to its environment *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)* Note: External attributes are those that relate to requirements (external properties of the software). External attributes can only be derived from the operational behavior of the system of which it is a part.

**external dependency.** (1) a relationship between project activities and non-project activities *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**external event.** (1) event from an external object, typically an interrupt from an external I/O device *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**external I/O device.** (1) hardware input and/or output device that is outside the software system and part of the external environment *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**external input (EI).** (1) a unique function recognized by the user in which data and/or control information from outside the application is entered into the application *(ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)* *(2)*

**external inquiry (EQ).** (1) a unique input/output combination recognized by the user in which the application distributes an output fully determined in size without further data processing, as a result of the input *(ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)* *(2)*

**external interface file (EIF).** (1) user-recognizable group of logically related data or control information, which is referenced by the application being measured, but which is maintained within the boundary of another application *(ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.29)* Note: The primary intent of an EIF is to hold data referenced through one or more ILFs and/or to alter the behavior of the system. An external input is a type of base functional component. See also: external inquiry, external output
be in an ILF in another application. An external interface file is a type of base functional component. See also: internal logical file, external logical file.

**external interface requirement.** (1) system or software requirement that specifies a hardware, software, or database element with which a system/software system or system/software component must interface, or that sets forth constraints on formats, timing, or other factors caused by such an interface. *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**external logical file.** (1) a logical group of permanent data seen from the perspective of the user that an application uses but that another application maintains. *(ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)*

**external measure.** (1) indirect measure of a product derived from measures of the behavior of the system of which it is a part. *(ISO/IEC 24765:2017 Systems and software engineering-Vocabulary)* Note: The number of failures found during testing is an external measure of the number of faults in the program, because the number of failures is counted during the operation of a computer system running the program. External measures can be used to evaluate quality attributes closer to the ultimate objectives of the design.

**external measure of software quality.** (1) measure of the degree to which a software product enables the behavior of a system to satisfy stated and implied needs for the system including the software to be used under specified conditions. *(ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.5)* Note: Attributes of the behavior can be verified or validated by executing the software product during testing and operation. See also: external software quality, internal measure of software quality.

**external measure of system or software quality.** (1) measure of the degree to which a system or software product enables the behavior to satisfy stated and implied needs for the system, including the software to be used under specified conditions. *(ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.11)* Note: Attributes of the behavior can be verified or validated by executing the system or software product during testing and operation.

**external output (EO).** (1) a unique output recognized by the user which crosses the application boundary. *(ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)* (2) elementary process that sends data or control information outside the application's boundary and includes additional processing logic beyond that of an external inquiry. *(ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method)* Note: The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information. The processing logic must contain at least one mathematical formula or calculation, or create derived data. An external output can also maintain one or more ILFs and/or alter the behavior of the system. An external output is a type of base functional component. See also: external input, external inquiry.

**external quality.** (1) extent to which a product satisfies stated and implied needs when used under specified conditions. *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*
external service provider. (1) organization providing services commercially to external customers (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.11)

external variability. (1) variability that is visible to customers (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.4)

ejective approach. (1) approach of developing the initial baseline of a product line from one or more existing products (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.13)

extranet. (1) set of intranets connected for specific objectives, spanning multiple organizations (ISO/IEC/IEEE 24765e:2015)


F1 score. (1) performance metric used to evaluate a classifier, which provides a balance (the harmonic average) between recall and precision (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.33) Syn: F1-score

facet. (1) operation interface in which a computational component plays a server role (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.16) (2) distinct named interface provided by the component for client interaction (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: the primary vehicle through which a component exposes its functional application behavior to clients during normal execution

categorized search. (1) progressive search which allows users to narrow the results by selecting values for one or more attributes (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.8)

facilitated workshops. (1) an elicitation technique using focused sessions that bring key cross-functional stakeholders together to define product requirements (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


factoring. (1) process of decomposing a system into a hierarchy of modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of removing a function from a module and placing it into a module of its own (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: modular decomposition

fail safe. (1) pertaining to a system or component that automatically places itself in a safe operating mode in the event of a failure (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: fail-safe, failsafe See also: fail soft, fault secure, fault tolerance

fail soft. (1) pertaining to a system or component that continues to provide partial operational capability in the event of certain failures (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: fail safe, fault secure, fault tolerance


failure mode. (1) physical or functional manifestation of a failure (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

failure mode and effect analysis (FMEA). (1) an analytical procedure in which each potential failure mode in every component of a product is analyzed to determine its effect on the reliability of that component and, by itself or in combination with other possible failure modes, on the reliability of the product or system and on the required function of the component; or the examination of a product (at the system or lower levels) for all ways that a failure may occur. For each potential failure, an estimate is made of its effect on the total system and of its impact. In addition, a review is undertaken of the action planned to minimize the probability of failure and to minimize its effects. (ISO/IEC/IEEE 24765:2019)

failure rate. (1) ratio of the number of failures of a given category to a given unit of measure (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: failure ratio

failure transparency. (1) distribution transparency which masks, from an object, the failure and possible recovery of other objects (or itself), to enable fault tolerance (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.4.1.2)

fallback plan. (1) an alternative set of actions and tasks available in the event the primary plan must be abandoned because of issues, risks or other causes (A Guide to the Project Management Body of Knowledge (PMBOK) Guide - Sixth Edition)

false negative. (1) true defect that has not been observed (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.8) (2) incorrect reporting of a failure when in reality it is a pass (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.34) See also: false positive, true negative

false positive. (1) observed defect which does not correspond to a true defect (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.9) (2) incorrect reporting of a pass when in reality it is a failure (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11:...
Guidelines on the testing of AI-based systems, 3.1.35) Note: False positives can appear during the analysis of an application due to lack of precision of the analysis rules. See also: false negative, true positive

families of programs. (1) sets of programs that are related by sharing significant portions of requirements, design, and code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: a program family might include one version of a program developed for an English-speaking audience, a second version of a program developed for a German-speaking audience, and a third version for a Japanese-speaking audience

family of platforms. (1) software platforms that are serving the same purpose (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1)

fast tracking. (1) a schedule compression technique in which activities or phases normally done in sequence are performed in parallel for at least a portion of their duration (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: schedule compression, crashing

fatal error. (1) error that results in the complete inability of a system or component to function (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


fault dictionary. (1) a list of faults in a system or component, and the tests that have been designed to detect them (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

fault isolation. (1) ability of a subsystem to prevent a fault within the subsystem from causing consequential faults in other subsystems (ISO/IEC/IEEE 24765c:2014)

fault masking. (1) condition in which one fault prevents the detection of another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

fault secure. (1) pertaining to a system or component in which no failures are produced from a prescribed set of faults (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: fault tolerance, fail-safe, fail soft

fault tolerance. (1) degree to which a system, product or component operates as intended despite the presence of hardware or software faults (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.5.3) (2) pertaining to the study of errors, faults, and failures, and of methods for enabling systems to continue normal operation in the presence of faults (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: error tolerance, fail safe, fail soft, fault secure, robustness
fault-tolerant. (1) pertaining to a system or component that is able to continue normal operation despite the presence of faults (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: fault tolerant

FCA. (1) functional configuration audit (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

FD. (1) full deployment (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

FDC. (1) functional domain categorization (ISO/IEC TR 14143-5:2004 Information technology -- Software measurement -- Functional size measurement -- Part 5: Determination of functional domains for use with functional size measurement, 4)

FDT. (1) formal description techniques (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

feasibility. (1) degree to which the requirements, design, or plans for a system or component can be implemented under existing constraints (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

feasibility study. (1) study to identify and analyze a problem and its potential solutions in order to determine their viability, costs, and benefits (ISO/IEC 2382:2015 Information technology -- Vocabulary)


feature branch. (1) branch created for developing a particular set of features (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The branch is typically not released but is collapsed back at some point to its parent branch.

feature engineering. (1) in machine learning, activity in which those attributes in the raw data that best represent the underlying relationships that should appear in the model are identified for use in the training data (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.36) Syn: feature selection

feature freeze. (1) period during which no new features are added to a specific branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: allows the branch to stabilize for a release.


feature set. (1) collection of items which contain the test conditions of the test item to be tested which can be collected from risks, requirements, functions, models, etc. (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 1) (2) logical subset of the test item(s) that could be treated independently of other feature sets in the subsequent test design activities (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.10)
fee. (1) representation of profit as a component of compensation to a seller (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

fetch. (1) to locate and load computer instructions or data from storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: move, store


field of application (of a specification). (1) properties the environment of the ODP system must have for the specification of that system to be used (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.1.2)

field programmable gate array (FPGA). (1) logic device designed to be programmed after it is acquired (ISO/IEC/IEEE 24765d:2015) Note: often based on look-up table architecture

fieldbus. (1) industrial computer network protocol used for real-time distributed control (ISO/IEC/IEEE 24765d:2015) Note: a family of related standardized interfaces

fifth-generation language (5GL). (1) computer language that incorporates the concepts of knowledge-based systems, expert systems, inference engines, and natural language processing (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembly language, fourth-generation language, high-order language, machine language

FIGS. (1) French, Italian, German, and Spanish (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)

figurative constant. (1) data name that is reserved for a specific constant in a programming language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: literal

file. (1) set of related records treated as a unit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) named set of records stored or processed as a unit (ISO/IEC 2382:2015 Information technology -- Vocabulary)

file type referenced (FTR). (1) data function read or maintained by a transactional function (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.31)

(2) an internal logical file (ILF) or an external logical file (ELF) maintained or read by a transaction (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: A file type referenced (FTR) includes an internal logical file read or maintained by a transactional function, or an external interface file read by a transactional function. Code data is grouped into one additional FTR.

final function point analysis. (1) an analysis to determine the functional size at the end of a project (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Note: The analysis can record the functional size of a specified or installed application, or can determine the size of the project implemented.

final transfer set. (1) collection of changed objects that are to be transferred integrally to one or more production
environments, including implementation instructions (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.18)

**financial independence. (1)** of software quality assurance (SQA), situation in which control of the SQA budget is vested in an organization independent of the development organization (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2)

**finish date. (1)** a point in time associated with a schedule activity’s completion. Usually qualified by one of the following: actual, planned, estimated, scheduled, early, late, baseline, target, or current. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition)

**finish-to-finish (FF). (1)** a logical relationship in which a successor activity cannot finish until a predecessor activity has finished (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition) See also: logical relationship

**finish-to-start (FS). (1)** a logical relationship in which a successor activity cannot start until a predecessor activity has finished (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition) See also: logical relationship

**finite state machine. (1)** computational model consisting of a finite number of states and transitions between those states, possibly with accompanying actions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**firm-fixed-price contract (FFP). (1)** a type of fixed price contract where the buyer pays the seller a set amount (as defined by the contract), regardless of the seller’s costs (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition) Syn: firm fixed price contract

**firmware. (1)** combination of a hardware device and computer instructions and data that reside as read-only software on that device (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (2) ordered set of instructions and associated data stored in a way that is functionally independent of main storage, usually in a ROM (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: The software cannot be readily modified under program control.

**first input routine. (1)** those activities required to obtain the logical record, if any, to be processed first (ISO/IEC/IEEE 24765a:2011)

**FiSMA. (1)** Finnish Software Measurement Association (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FiSMA 1.1 functional size measurement method, A.10) Note: a network of Finnish companies, which share interest in developing software measurement and/or software processes.

**fixed cost. (1)** cost that is not dependent on the rate of production (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: such as facilities cost or loan interest See also: variable cost

**fixed formula method. (1)** earned value method for assigning a specified percentage of budget value of a work package to the start milestone of the work package with the remaining budget value percentage assigned when the work package is complete (ISO/IEC/IEEE 24765h:2019)

**fixed price with economic price adjustment contract (FP-EPA). (1)** A fixed-price contract, but with a special provision allowing for pre-defined final adjustments to the contract price due to changed conditions, such as inflation changes, or cost increases (or decreases) for specific commodities. (A Guide to the Project Management Body of
fixed-cost analysis. (1) analysis that seeks to maximize the effectiveness that can be attained from a fixed, maximum investment (ISO/IEC/IEEE 24765:2017 Systems and software engineering–Vocabulary) See also: fixed-effectiveness analysis

fixed-effectiveness analysis. (1) analysis that seeks to minimize the investment needed to attain a fixed, minimum degree of effectiveness (ISO/IEC/IEEE 24765:2017 Systems and software engineering–Vocabulary) See also: fixed-cost analysis

fixed-price contract. (1) an agreement that sets the amount that will be paid for a defined scope of work regardless of cost or effort to deliver it (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: fixed price contract, fixed-price contracts

fixed-price-incentive-fee (FPIF) contract. (1) a type of contract where the buyer pays the seller a set amount (as defined by the contract), and the seller can earn an additional amount if the seller meets defined performance criteria. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: fixed price incentive fee contract

flag. (1) variable that is set to a prescribed state, often 'true' or 'false,' based on the results of a process or the occurrence of a specified condition (ISO/IEC/IEEE 24765:2017 Systems and software engineering–Vocabulary) See also: indicator, semaphore

flash memory. (1) larger and faster programmable ROM which allows data to be electrically erased from memory and rewritten many times (ISO/IEC/IEEE 24765c:2014) Syn: NVRAM, non-volatile random access memory See also: EEPROM

flexibility. (1) ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed (ISO/IEC/IEEE 24765:2017 Systems and software engineering–Vocabulary) (2) degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in contexts beyond those initially specified in the requirements (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.5.2) (3) degree to which a product or system can be used with acceptable levels of effectiveness, efficiency, freedom from risk and satisfaction in contexts beyond those initially specified in the requirements (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.7) (4) ability of a system to work in contexts outside its initial specification (i.e. change its behavior according to its actual situation to satisfy its objectives) (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.37) Note: Flexibility enables products to take account of circumstances, opportunities and individual preferences that had not been anticipated in advance. If a product is not designed for flexibility, it might not be safe to use the product in unintended contexts. Flexibility can be measured either as the extent to which a product can be used by additional types of users to achieve additional types of goals with effectiveness, efficiency, freedom from risk and satisfaction in additional types of contexts of use, or by a capability to be modified to support adaptation for new types of users, tasks and environments, and suitability for individualization. See also: adaptability, extendability, maintainability
flip-flop. (1) electronic circuit with one or two stable states (ISO/IEC/IEEE 24765c:2014) Note: can be used to store 0 or 1 as digital data Syn: latch

float. (1) amount of unscheduled time between sequential activities not on the critical path, which can be used to delay the completion of the earlier activity or advance the start of the later activity (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: slack See also: free float, total float

flow. (1) abstraction of a sequence of interactions, resulting in conveyance of information from a producer object to a consumer object (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.) Note: A flow can be used to abstract over, for example, the exact structure of a sequence of interactions, or over a continuous interaction including the special case of an analogue information flow.

flowchart. (1) graphical representation of a process or the step-by-step solution of a problem, using suitably annotated geometric figures connected by flowlines for the purpose of designing or documenting a process or program (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) graphical representation of the definition, analysis, or method of solution of a problem in which symbols are used to represent operations, data, flow, equipment, etc (ISO 5807:1985 Information processing -- Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts, 3.3) (3) control flow diagram in which suitably annotated geometrical figures are used to represent operations, data, or equipment, and arrows are used to indicate the sequential flow from one to another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) the depiction in a diagram format of the inputs, process actions, and outputs of one or more processes within a system (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: flow chart, flow diagram See also: block diagram, box diagram, bubble chart, graph, input-process-output chart, structure chart

flowcharter. (1) software tool that accepts as input a design or code representation of a program and produces as output a flowchart of the program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

FMEA. (1) failure mode and effect analysis (ISO/IEC/IEEE 24765h:2019)

FMECA. (1) failure mode, effects, and criticality analysis (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

FOC. (1) full operational capability (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

focus groups. (1) an elicitation technique that brings together prequalified stakeholders and subject matter experts to learn about their expectations and attitudes about a proposed product, service, or result (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

footer. (1) material repeated at the bottom of each page (ISO/IEC/IEEE 24765a:2011)

For Exposition Only (FEO) page. (1) model page that contains pictorial and graphical information (in contrast to text) about a specific diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.51) Note: Unlike a diagram, the contents of a For Exposition Only page (FEO page) need not comply with IDEF0 rules.

forecast. (1) estimate or prediction of conditions and events in the project's future based on information and knowledge available at the time of the forecast (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
**Edition) Note:** The forecast is based on the project’s past performance and expected future performance, and includes information that could impact the project in the future, such as estimate at completion and estimate to complete.

**foreground. (1)** In job scheduling, the computing environment in which high-priority processes or those requiring user interaction are executed *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: background, foreground processing

**foreground processing. (1)** Execution of a high-priority process while lower priority processes await the availability of computer resources, or the execution of processes that require user interaction *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: background processing

**foreign key. (1)** Attribute, or combination of attributes, of a child or category entity instance whose values match those in the primary key of a related parent or generic entity instance *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.62)* Note: A foreign key results from the migration of the parent or generic entity's primary key through a generalization structure or a relationship. [key style] Syn: migrated key

**forking action. (1)** Dividing action, where the enabled chains must (subject to failure) eventually join each other, i.e., the enabled chains cannot join other chains and they cannot terminate separately *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.5)*

**form. (1)** Module or formulary to collect data *(ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.20)* Note: It can be paper-based (paper form) or digital.

**form, fit, and function. (1)** In configuration management, that configuration comprising the physical and functional characteristics of an item as an entity, but not including any characteristics of the elements making up the item *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: configuration identification

**formal design. (1)** Process of using a formal method for software design *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**formal evaluation process. (1)** Structured approach to evaluating alternative solutions against established criteria to determine a recommended solution to address an issue *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**formal language. (1)** Language whose rules are explicitly established prior to its use *(ISO/IEC 2382:2015 Information technology -- Vocabulary)* Syn: artificial language See also: natural language

**formal parameter. (1)** Variable used in a software module to represent data or program elements that are to be passed to the module by a calling module *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: argument (3)

**formal qualification review (FQR). (1)** Test, inspection, or analytical process by which a group of configuration items comprising a system is verified to have met specific contractual performance requirements *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: code review, design review, requirements review, test readiness review
formal review. (1) form of review that follows a defined process with formal documented output (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.5)

formal specification. (1) specification that is used to prove mathematically the validity of an implementation or to derive mathematically the implementation (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) specification written in a formal notation, often for use in proof of correctness (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) specification written and approved in accordance with established standards (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

formal testing. (1) testing conducted in accordance with test plans and procedures that have been reviewed and approved by a customer, user, or designated level of management (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: informal testing

formal verification. (1) activity proving or disproving the correctness of intended applications with respect to a formal specification or a property, using formal methods of mathematics (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.10)

formalization. (1) precise description of the semantics of a language in terms of a formal language such as first order logic (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.63)

formative construct. (1) construct that is formed from its observed measures in the relationship between a construct and its measures (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.7) Note: The construct is a consequence of its measures and each measure is a determinant of the construct.

formative evaluation. (1) evaluation designed and used to improve the object of evaluation, especially when it is still being developed (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.8)

forward pass. (1) a critical path method technique for calculating the early start and early finish dates by working forward through the schedule model from the project start date or a given point in time (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: schedule network analysis, backward pass

forward recovery. (1) reconstruction of a file to a given state by updating an earlier version, using data recorded in a chronological record of changes made to the file (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) type of recovery in which a system, program, database, or other system resource is restored to a new, not previously occupied state in which it can perform required functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

FOSS. (1) free and open source software (ISO/IEC/IEEE 24765f:2016)

four-address instruction. (1) computer instruction that contains four address fields (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-address instruction, two-address instruction, three-address
four-plus-one address instruction. (1) computer instruction that contains five address fields, the fifth containing the address of the instruction to be executed next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-plus-one address instruction, two-plus-one address instruction, three-plus-one address instruction

fourth-generation language (4GL). (1) computer language designed to improve the productivity achieved by high-order (third-generation) languages and, often, to make computing power available to non-programmers (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Features typically include an integrated database management system, query language, report generator, screen definition facility, graphics generator, decision support function, financial modeling, spreadsheet capability, and statistical analysis functions. See also: machine language, assembly language, high order language, fifth-generation language

FP. (1) function point (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)


FPA. (1) function point analysis (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 4) (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

FPA table. (1) an entity type that has a secondary function in the application (e.g., code tables, reference tables, entity types with constants, text, or decodings) and whose data can be maintained by the application to be counted or by a different application (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

FPA tables ELF. (1) external logical file that is counted for the set of all FPA tables identified in an application that are only used by the application to be counted, but that are maintained by another application (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Syn: function point analysis tables external logical file

FPA tables ILF. (1) the internal logical file that is counted for the set of all identifiable and maintainable FPA tables in an application (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

FPGA. (1) field programmable gate array (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)


FQR. (1) formal qualification review (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

FRACAS. (1) failure reporting and corrective action system (IEEE 15288:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

frame. (1) mechanism for dividing a browser window into independent windows for displaying different content or different parts of the same content (document) (ISO/IEC/IEEE 23026:2015 Systems and software engineering--
Engineering and management of websites for systems, software, and services information, 4.10)

framework. (1) reusable design (models or code) that can be refined (specialized) and extended to provide some portion of the overall functionality of many applications (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.64) (2) partially completed software subsystem that can be extended by appropriately instantiating some specific plug-ins (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) for a component content management system, essential data structures, operations, and rules that form the foundation from which all other features of the CCMS are built (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.19)

free float. (1) the amount of time that a schedule activity can be delayed without delaying the early start date of any successor or violating a schedule constraint (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: total float

freedom from risk. (1) degree to which a product or system mitigates the potential risk to economic status, human life, health, or the environment (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.4) (2) degree to which the quality of a product or system mitigates or avoids the potential risk to economic status, human life, health, or the environment (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.9) Note: Freedom from risk includes reduction of potential risks to the user, organization or project. See also: risk

front matter. (1) material that comes at the front of a printed book or manual, such as the title page and table of contents (ISO/IEC/IEEE 24765a:2011)

frozen branch. (1) branch where no development takes place, either in preparation for a release or because active development has ceased on it (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

FRP. (1) full-rate production (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

FRR. (1) flight readiness review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)


FSM method. (1) a specific implementation of FSM defined by a set of rules (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.4) Syn: FSMM


FTP. (1) File Transfer Protocol (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and
management of websites for systems, software, and services information, 5)

FTR. (1) file type referenced (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 4) (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

full duplex. (1) able to communicate data in both directions simultaneously (ISO/IEC/IEEE 24765d:2015) See also: half duplex

function. (1) a task, action, or activity that must be accomplished to achieve a desired outcome. (2) defined objective or characteristic action of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) software module that performs a specific action, is invoked by the appearance of its name in an expression, receives input values, and returns a single value (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) part of an application that provides facilities for users to carry out their tasks (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.21) (5) elementary unit of requirements and specifications defined and used for measurement purposes (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (6) single-valued mapping (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.65) (7) transformation of inputs to outputs, by means of some mechanisms, and subject to certain controls, that is identified by a function name and modeled by a box (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.53)

function name. (1) active verb or verb phrase that describes what is to be accomplished by a function (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.54) Note: A box takes as its box name the function name of the function represented by the box.

function point (FP). (1) a unit which expresses the size of an application or of a project (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) unit of measurement to express the amount of functionality an information system (as a product) provides to a user (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) (3) a measure of the delivered software functionality (IEEE 1045-1992, (R2002) IEEE Standard for Software Productivity Metrics, 3.2)

function point analysis (FPA). (1) a method used to acquire a measurement of the amount of functionality an application provides a user (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) a form of functional size measurement (FSM) that measures the functional size of software development, enhancement and maintenance activities associated with business applications, from the customer’s point of view (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) (3) method for measuring functional size as defined in ISO/IEC 29026:2009 (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.36)

function point count. (1) activity of applying rules to measure the functional size of an application or project

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
function point table. (1) the table used to allocate function points to functions, depending on the function type and the complexity established for the function. (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

function type. (1) the five types of components in two categories (data function types and transactional function types) of which an application consists, seen from the perspective of the FPA method. (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) type of base functional component identified in ISO/IEC 20926:2009. (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.38) Note: The five function types are External Input, External Output, External Inquiry, Internal Logical File and External Interface File or External Logical File.

function-oriented design. (1) partitioning of a design into subsystems and modules, with each one handling one or more functions. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Software and Systems Engineering Vocabulary) See also: object-oriented design, data-structure-oriented design

functional analysis. (1) systematic investigation of the functions of a real or planned system. (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) examination of a defined function to identify all the subfunctions necessary to accomplish that function, to identify functional relationships and interfaces (internal and external) and capture these in a functional architecture, to flow down upper-level performance requirements and to assign these requirements to lower-level subfunctions. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

functional appropriateness. (1) degree to which the functions facilitate the accomplishment of specified tasks and objectives. (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.1.3) Note: Functional appropriateness corresponds to suitability for the task.

functional architecture. (1) arrangement of functions and their subfunctions and interfaces (internal and external) that defines the execution sequencing, conditions for control or data flow, and the performance requirements to satisfy the requirements baseline. (ISO/IEC/IEEE 24765:2016) (2) hierarchical arrangement of functions, their internal and external (external to the aggregation itself) functional interfaces and external physical interfaces, their respective functional and performance requirements, and their design constraints. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

functional baseline. (1) description of the system's performance (functional, interoperability, and interface characteristics) and the verification required to demonstrate the achievement of those specified characteristics. (IEEE 15288:1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1) See also: allocated baseline, developmental configuration, product baseline

functional cohesion. (1) type of cohesion in which the tasks performed by a software module all contribute to the performance of a single function. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also:
coincidental cohesion, communicational cohesion, logical cohesion, procedural cohesion, sequential cohesion, temporal cohesion

**functional completeness.** (1) degree to which the set of functions covers all the specified tasks and user objectives *(ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.1.1)*

**functional complexity.** (1) specific complexity rating assigned to a function using defined rules *(ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.32)*

**functional configuration audit (FCA).** (1) audit conducted to verify that the development of a configuration item has been completed satisfactorily, that the item has achieved the performance and functional characteristics specified in the functional or allocated configuration identification, and that its operational and support documents are complete and satisfactory *(IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)* *(2) evaluation to ensure that the product meets baseline functional and performance capabilities *(INCOSE Systems Engineering Handbook, 4th ed.)* See also: configuration management, physical configuration audit

**functional configuration identification.** (1) in configuration management, the current approved technical documentation for a configuration item *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It prescribes all necessary functional characteristics, the tests required to demonstrate achievement of specified functional characteristics, the necessary interface characteristics with associated configuration items, the configuration item's key functional characteristics and its key lower-level configuration items, if any, and design constraints. See also: allocated configuration identification, product configuration identification functional baseline*

**functional correctness.** (1) degree to which a product or system provides the correct results with the needed degree of precision *(ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.1.2)*

**functional decomposition.** (1) type of modular decomposition in which a system is broken down into components that correspond to system functions and subfunctions *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in software engineering, the partitioning of higher-level system functions into smaller and smaller pieces to render them more manageable and understandable See also: hierarchical decomposition, stepwise refinement

**functional design.** (1) process of defining the working relationships among the components of a system *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) result of the process in (1) *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) specification of the functions of the components of a system and of the working relationships among them *(ISO/IEC 2382:2015 Information technology -- Vocabulary)* See also: architectural design

**functional domain.** (1) class of software based on the characteristics of functional user requirements which are pertinent to FSM *(ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.5) (2) categorized functions that are generally used together *(ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.16)*

**functional domain categorization (FDC).** (1) process for identifying functional domains *(ISO/IEC TR 14143-
functional language. (1) programming language used to express programs as a sequence of functions and function calls (ISO/IEC/IEEE 24765:2017 Systems and software engineering -- Vocabulary)

functional manager. (1) someone with management authority over an organizational unit within a functional organization, the manager of any group that actually makes a product or performs a service (ISO/IEC/IEEE 24765h:2019) Syn: line manager

functional organization. (1) An organizational structure in which staff is grouped by areas of specialization and the project manager has limited authority to assign work and apply resources. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

functional process. (1) elementary component of a set of functional user requirements, comprising a unique, cohesive and independently executable set of data movements (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.10) (2) an elementary component of a set of Functional User Requirements, comprising a unique, cohesive and independently executable set of data or data movements (functional services) (ISO/IEC 29881:2010 Information technology -- Software and systems engineering -- FiSMA 1.1 functional size measurement method, A.11) Note: It is triggered by a data movement (an Entry) from a functional user that informs the piece of software that the functional user has identified a triggering event, and is complete when it has executed all that is required to be done in response to the triggering event. Syn: functional process type, transactional process

functional requirement. (1) statement that identifies what results a product or process shall produce (ISO/IEC/IEEE 24765f:2016) (2) requirement that specifies a function that a system or system component shall perform (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) See also: nonfunctional requirement

functional service. (1) base functional component (BFC) (ISO/IEC 29881:2010 Information technology -- Software and systems engineering -- FiSMA 1.1 functional size measurement method, 3.6) (2) service that must be implemented in the piece of software in order to fulfill functional user requirements (ISO/IEC 29881:2010 Information technology -- Software and systems engineering -- FiSMA 1.1 functional size measurement method, A.9)


functional specification. (1) document that specifies the functions that a system or component must perform (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: often part of a requirements specification

functional suitability. (1) degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions (ISO/IEC 25010:2011 Systems and software engineering--Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.1) Note: Functional suitability is only concerned with whether the functions meet stated and implied needs, not the functional specification.

functional system design. (1) specification of the functions of the components of a software system and of the working relationships between them (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.19)

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 (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) testing conducted to evaluate the compliance of a system or component with specified functional requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: black-box testing See also: structural testing, specification-based testing

functional unit. (1) entity of hardware or software, or both, capable of accomplishing a specified purpose (ISO/IEC 2382:2015 Information technology -- Vocabulary)

functional user. (1) user that is a sender or an intended recipient of data in the Functional User Requirements of a piece of software (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.13)

user requirements (FUR). (1) subset of the user requirements specifying what the software shall do in terms of tasks and services (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.34) (2) a subset of the user requirements describing what the software does in terms of tasks and services (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.14) (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.8) Note: Functional User Requirements include but are not limited to: data transfer (for example Input customer data, Send control signal); data transformation (for example Calculate bank interest, Derive average temperature); data storage (for example Store customer order, Record ambient temperature over time); data retrieval (for example List current employees, Retrieve aircraft position). User Requirements that are not Functional User Requirements include but are not limited to: quality constraints (for example usability, reliability, efficiency and portability); organizational constraints (for example locations for operation, target hardware and compliance to standards); environmental constraints (for example interoperability, security, privacy and safety); implementation constraints (for example development language, delivery schedule).

functionality. (1) capabilities of the various computational, user interface, input, output, data management, and other features provided by a product (ISO/IEC/IEEE 24765e:2015) Note: This characteristic is concerned with what the software does to fulfill needs. The software quality characteristic functionality can be used to specify or evaluate the suitability, accuracy, interoperability, security, and compliance of a function.
limits on the commitment of funds for the project to identify any variances between the funding limits and the planned expenditures (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


fuse ROM. (1) programmable ROM with written data based on a fuse connection state (ISO/IEC/IEEE 24765c:2014)

future worth. (1) representation of a cash flow as a single instance at the end of the planning horizon (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: annual equivalent, present worth

fuzz testing. (1) software testing approach in which high volumes of random (or near-random) data, called fuzz, are used to generate inputs to the test item (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.38)

Gantt chart. (1) a bar chart of schedule information where activities are listed on the vertical axis, dates are shown on the horizontal axis, and activity durations are shown as horizontal bars placed according to start and finish dates (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: bar chart

garbage collection. (1) in computer resource management, a synonym for memory compaction (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


general artificial intelligence. (1) artificial intelligence that exhibits intelligent behavior comparable to a human across the full range of cognitive abilities (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.39) Syn: general AI, strong artificial intelligence, strong AI

general purpose input/output port (GPIO). (1) generic pin (port) on a microcomputer whose function (whether it is an input or output pin) is not predefined and is user-controlled (ISO/IEC/IEEE 24765d:2015) Syn: general-purpose input output port

general register. (1) register that stores both addresses and data (ISO/IEC/IEEE 24765e:2015)

general system characteristics (GSCs). (1) terminology for technical complexity adjustment factors (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

generality. (1) degree to which a system or component performs a broad range of functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: reusability

generalization. (1) taxonomy in which instances of both entities represent the same real or abstract thing (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.66) Note: One entity (the generic entity) represents the complete set of things and the other (category entity) represents a subtype or sub-classification of those things. The category entity can have one or more attributes, or relationships with instances of another entity, not shared by all generic entity instances. Each instance of the category entity is simultaneously an instance of the generic entity. [key style] See also: categorization

generalization structure. (1) connection between a superclass and one of its more specific, immediate subclasses

---

This definition is copyrighted ©, 2021 by the IEEE.

The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition.

All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.

Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.

Note: In a generalization taxonomy every instance is fully described by one or more of the classes in the taxonomy. The structuring of classes as a generalization taxonomy determines the inheritance of responsibilities among classes. Syn: generalization hierarchy, generalization network

generalize. (1) saying that a subclass s generalizes to a superclass C means that every instance of class s is also an instance of class C. (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.66) Note: Generalization is fundamentally different from a relationship, which can associate distinct instances.

generally accepted. (1) knowledge to be included in the study material of a software engineering licensing exam that a graduate would pass after completing four years of work experience. (ISO/IEC TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK))

generated address. (1) address that has been calculated during the execution of a computer program. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: synthetic address See also: absolute address, effective address, relative address, indirect address

generation. (1) act of defining and describing a methodology from a particular metamodel. (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.8) Note: Generating a methodology includes explaining the structural position and semantics of each methodology element using the selected metamodel. Thus, what methodology elements are possible, and how they relate to each other, are constrained by such a metamodel. Usually, method engineers perform generation, yielding a complete and usable methodology.

generic ancestor (of a class). (1) superclass that is either an immediate superclass of the class or a generic ancestor of one of the superclasses of the class. (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.71) See also: reflexive ancestor

generic entity. (1) entity whose instances are classified into one or more subtypes or subclassifications (category entities). (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.72) Note: [key style] See also: superclass, supertype

generic practice. (1) activity that, when consistently performed, contributes to the achievement of a specific process attribute. (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.6)

generic profile group. (1) profile group applicable to very small entities (VSEs) that do not develop critical systems or software products and have typical situational factors. (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.9) (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.28)

generic program unit. (1) software module that is defined in a general manner and that requires substitution of specific data, instructions, or both, in order to be used in a computer program. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: instantiation
GFE. (1) government-furnished equipment (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

GFI. (1) government-furnished information (ISO/IEC/IEEE 24765j:2021)


glass box. (1) system or component whose internal contents or implementation are known (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to an approach that treats a system or component as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: white box See also: black box

global attribute. (1) condition when the attributes that describe the foreign keys are the same attributes (and attribute values) as those describing the associated candidate key (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.10)

global compaction. (1) in microprogramming, compaction in which microoperations can be moved beyond the boundaries of the single-entry, single-exit sequential blocks in which they occur (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: white box See also: local compaction

global data. (1) data that can be accessed by two or more non-nested modules of computer program without being explicitly passed as parameters between the modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: common data See also: local data

global label. (1) label associated with the net graph itself, rather than with an object of a net graph (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.4)

global navigation. (1) set of navigation links available on all pages of a website (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.11)

global variable. (1) variable that can be accessed by two or more non-nested modules of a computer program without being explicitly passed as a parameter between the modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: common data See also: local data

globally unique identifier (GUID). (1) 16-byte string of characters that is generated in a manner that gives a high probability that the string is unique in any context (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.16) Note: GUID as an all capitalized term refers specifically to the 16-byte version. If the term is in lowercase (guid), it refers to a general algorithm that can use either a URI or a 16-byte-based identifier.

glossary. (1) collection of the names and narrative descriptions of all terms that can be used for defined concepts within an environment (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.73) (2) a set of definitions that includes arrow labels and box names used in an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics
for IDEF0, 2.1.55)
glossary page. (1) model page that contains definitions for the arrow labels and box names in a specific diagram
GO. (1) governance process (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 4.1)
go to. (1) computer program statement that causes a jump (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: call, case, if-then-else branch
GOTS. (1) government-off-the-shelf (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)
Government-off-the-Shelf. (1) software supplied by the government for reuse in another project (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: GOTS, Government-Off-The-Shelf, Government off the Shelf, Government Off The Shelf See also: COTS
GPIO. (1) general-purpose input/output port (ISO/IEC/IEEE 24765d:2015)
grade. (1) category or rank used to distinguish items that have the same functional use, but do not share the same requirements for quality (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
granularity. (1) depth or level of detail at which data is collected (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
graph. (1) diagram that represents the variation of a variable in comparison with that of one or more other variables (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) diagram or other representation consisting of a finite set of nodes and internode connections called edges or arcs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
graphical information. (1) information defining the graphical appearance of objects and labels of a net graph, which can be the position, size, line color, fill color, font, or line width (ISO/IEC 15909-2:2011 Software and system engineering - High-level Petri nets -- Part 2: Transfer format, 4.1.3)
graphical processing unit (GPU). (1) application-specific integrated circuit (ASIC) specialized for display
functions such as rendering images (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.40) Note: GPUs are designed for parallel data processing of images with a single function, and this parallel processing is also useful for executing artificial intelligence software, such as neural networks.

graphical symbol. (1) visually perceptible figure with a particular meaning used to transmit information independently of language (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.13)

Grosch's law. (1) guideline formulated by H. R. J. Grosch, stating that the computing power of a computer increases proportionally to the square of the cost of the computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

ground rules. (1) list of acceptable and unacceptable behaviors adopted by a project team to improve working relationships, effectiveness, and communication (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) expectations regarding acceptable behavior by project team members (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

group. (1) number of model elements regarded as a unit formed by traceability relationships to a single distinct element (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)


GUID. (1) globally unique identifier (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.16)


guideline. (1) official recommendation or advice that indicates policies, standards, or procedures for how something should be accomplished (ISO/IEC/IEEE 24765h:2019) See also: guide, standard

gull wing lead. (1) connector from the thin side of an integrated circuit package which extends out, down, and then out horizontally to allow it to be connected within a device (ISO/IEC/IEEE 24765c:2014) Note: The bent shape is thought to resemble a bird wing.

hacker. (1) technically sophisticated computer enthusiast (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) technically sophisticated computer enthusiast who uses his or her knowledge and means to gain unauthorized access to protected resources (ISO/IEC 2382:2015 Information technology -- Vocabulary)

half duplex. (1) able to communicate data in both directions, but in only one direction at a time (ISO/IEC/IEEE 24765d:2015) See also: full duplex

Audits on Defense Programs, 3.2)

**hard copy.** (1) permanent copy of a display image generated on an output unit such as a printer or a plotter, and which can be carried away (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**hard failure.** (1) failure that results in complete shutdown of a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: soft failure

**hardware.** (1) physical equipment used to process, store, or transmit computer programs or data (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.21) (2) all or part of the physical components of an information system (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: HW See also: software

**hardware configuration item (HCI).** (1) aggregation of hardware that is designated for configuration management and treated as a single entity in the configuration management process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: An HCI exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the hardware has been established as a configuration item. See also: software configuration item

**hardware description language (HDL).** (1) software programming language used to design and model hardware, especially digital logic circuits (ISO/IEC/IEEE 24765d:2015) Syn: hardware design language

**hardware engineering.** (1) application of a systematic, disciplined, and quantifiable approach to design, implement, and maintain a tangible product by transforming a set of requirements that represent the collection of stakeholder needs, expectations, and constraints; using documented techniques and technology (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: software engineering, systems engineering

**hardware monitor.** (1) device that measures or records specified events or characteristics of a computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) software tool that records or analyzes hardware events during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**harm.** (1) injury or damage to the health of people, or damage to property or the environment (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.14)

**Harvard architecture.** (1) computer architecture with physically separate communication paths for instructions and data (ISO/IEC/IEEE 24765d:2015)

**hazard.** (1) intrinsic property or condition that has the potential to cause harm or damage (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.11) (2) source of potential harm or a situation with a potential for harm in terms of human injury; damage to health, property, or the environment; or some combination of these (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.11) (3) source of potential harm (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.17) (4) potential source of harm (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.15)

**hazard identification.** (1) process of recognizing that a hazard exists and defining its characteristics (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.12)

HDD. (1) hardware design description (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)

HDL. (1) hardware description language (ISO/IEC/IEEE 24765d:2015) (2) hardware design language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: design language

HDTV. (1) High Definition TV (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

head. (1) forefront of a branch, which contains the evolving versions of the source tree (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A release coming out of head will have the newest features but will also likely be unstable.

head action. (1) in a given activity, an action that has no predecessor (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.7)

header. (1) block of comments placed at the beginning of a computer program or routine (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) identification or control information placed at the beginning of a file or message (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) material repeated at the top of each page (ISO/IEC/IEEE 24765a:2011)

heading. (1) text that identifies the topic that will be covered in the following text (ISO/IEC/IEEE 24765a:2011)

health and safety risk mitigation. (1) degree to which a product or system mitigates the potential risk to people in the intended contexts of use (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.4.2)

heavyweight process. (1) process with its own memory and multiple threads of control (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

help system. (1) ancillary part of a program, or sometimes a separate program, that allows the user to view parts of the online documentation or help text on request (ISO/IEC/IEEE 24765a:2011) See also: online documentation system

heuristic evaluation. (1) assessment by one or more experts who judge conformance to a recognized set of principles (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.18)


hierarchical decomposition. (1) type of modular decomposition in which a system is broken down into a hierarchy of components through a series of top-down refinements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: functional decomposition, stepwise refinement

hierarchical modeling. (1) technique used in computer performance evaluation, in which a computer system is represented as a hierarchy of subsystems, the subsystems are analyzed to determine their performance characteristics, and the results are used to evaluate the performance of the overall system (ISO/IEC/IEEE 24765:2017 Systems and
hierarchically consecutive. (1) an unbroken unidirectional traversal of all nodes between two specified nodes in a tree (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.57)

hierarchy. (1) a structure in which components are ranked into levels of subordination; each component has zero, one, or more subordinates; and no component has more than one superordinate component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) arrangement of model elements according to traceability relationships, where an element that owns or groups other elements is considered at a higher level than the owned (grouped) elements (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) See also: hierarchical decomposition, hierarchical modeling

high level. (1) general; abstract (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

high level function point analysis. (1) a possible function point analysis in an early phase of an application's life cycle to determine the size of an application or a project in which certain minimum specifications are assumed (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Note: Typically, the number of functions is recorded per type, and a default value is used for the complexity: average for the transactional functions (transactions) and low for the data functions (logical files). Syn: estimated function point analysis, high-level function point analysis See also: detailed function point analysis, indicative function point analysis

high-level design. (1) process of defining the high-level concepts that guide low-level design and implementation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: High-level design typically involves organizing a system into subprograms and specifying the interfaces between them. See also: architecture

high-level keyword. (1) keyword that covers complex activities that may be composed from other keywords and is used by domain experts to assemble keyword test cases (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 4.2)

high-order language (HOL). (1) programming language that requires little knowledge of the computer on which a program will run, can be translated into several different machine languages, allows symbolic naming of operations and addresses, provides features designed to facilitate expression of data structures and program logic, and usually results in several machine instructions for each program statement (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: high-level language, high order language, higher order language, third-generation language See also: assembly language, fifth-generation language, fourth-generation language, machine language

higher-level management. (1) person or persons who provide the policy and overall guidance for the process, but do not provide the direct day-to-day monitoring and controlling of the process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Such persons belong to a level of management in the organization above the immediate level responsible for the process.

histogram. (1) a special form of bar chart used to describe the central tendency, dispersion, and shape of a statistical distribution (ISO/IEC/IEEE 24570:2019) (2) a bar chart that shows the graphical representation of numerical data (A
historical information. (1) documents and data on prior projects including project files, records, correspondence, closed contracts, and closed projects. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

HLL. (1) high-level language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: high-order language

HMI. (1) human-machine interface (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: user interface

HOL. (1) high-order language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

home page. (1) page of a website through which users typically enter the website, and whose URL is typically published or linked as the main web address of the site or organization (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.12) Syn: center page, front page, index page, main page, start page, top page

homogeneous redundancy. (1) in fault tolerance, realization of the same function with identical means (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: diversity

horizontal microinstruction. (1) microinstruction that specifies a set of simultaneous operations needed to carry out a given machine language instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Horizontal microinstructions are relatively long, often 64 bits or more, and are called 'horizontal' because the set of simultaneous operations that they specify are written on a single line, rather than being listed sequentially down the page. See also: diagonal microinstruction, vertical microinstruction

host machine. (1) the computer on which a program or file is installed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in a computer network, a computer that provides processing capabilities to users of the network (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) computer used to develop software intended for another computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) computer used to emulate another computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

hostile backout. (1) backout done without prior arrangement by a committer other than the one who introduced the original change (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This is usually the opening shot in a commit war.

housekeeping operation. (1) computer operation that establishes or reestablishes a set of initial conditions to facilitate the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: overhead operation

HREF. (1) HTML reference designator (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

HRS. (1) hardware requirements specification (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)

HSI. (1) human systems integration (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)
HTML. (1) HyperText Markup Language (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

HTTP. (1) HyperText Transfer Protocol (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

human factors. (1) systematic application of relevant information about human abilities, characteristics, behavior, motivation, and performance (INCOSE Systems Engineering Handbook, 4th ed.) Note: It includes principles and applications in the areas of human-related engineering, anthropometrics, ergonomics, job performance skills and aids, and human performance evaluation.

human resource planning. (1) identification and documentation of project roles, responsibilities and reporting relationships, as well as estimation of required staff by time period and creation of a staffing management plan (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

human systems engineering. (1) activities involved throughout the system life cycle that address the human element of system design (including usability, measures of effectiveness, measures of performance, and total ownership cost) (ISO/IEC/IEEE 24765f:2016) Note: These activities include the definition and synthesis of manpower, personnel, training, human engineering, health hazards, and safety issues.


human-centered design. (1) approach to system design and development that aims to make interactive systems more usable by focusing on the use of the system; applying human factors, ergonomics and usability knowledge and techniques (ISO/IEC 25063:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description, 3.6) Note: The term "human-centered design" is used rather than "user-centered design" to emphasize that design impacts a number of stakeholders, not just those typically considered as users. However, in practice, these terms are often used synonymously. Usable systems can provide a number of benefits, including improved productivity, enhanced user well-being, avoidance of stress, increased accessibility, and reduced risk of harm. Syn: human-centred design, user-centered design, user-centred design

Hurwicz criterion. (1) in decision making under uncertainty, a method which gives each decision a value which is a weighted sum of its worst and best possible outcomes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: allows the decision maker to account for optimistic and pessimistic views See also: maximax rule, maximin rule, minimax regret rule

HW. (1) hardware (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)

HWCI. (1) hardware configuration item (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

hybrid computer. (1) computer that integrates analog computer components and digital computer components by interconnection of digital-to-analog converters and analog-to-digital converters (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: A hybrid computer can use or produce analog data and discrete data.
hybrid coupling. (1) type of coupling in which different subsets of the range of values that a data item can assume are used for different and unrelated purposes in different software modules. See also: common-environment coupling, content coupling, control coupling, data coupling, pathological coupling

hyperparameter. (1) variable used to define the structure of a neural network and how it is trained. Note: Typically, hyperparameters are set by the developer of the model and may also be referred to as a tuning parameter.

hypertext markup language (HTML). (1) language for creating web pages. See also: common-environment coupling, content coupling, control coupling, data coupling, pathological coupling

HyperText Transfer Protocol (HTTP). (1) application-level protocol for distributed, collaborative, hypermedia information systems. See also: common-environment coupling, content coupling, control coupling, data coupling, pathological coupling

I/O. (1) input/output

I/O task-structuring criteria. (1) category of the task-structuring criteria addressing how device interface objects are mapped to I/O tasks and when an I/O task is activated. See also: common-environment coupling, content coupling, control coupling, data coupling, pathological coupling

I2C. (1) inter-integrated circuit bus

IaaS. (1) infrastructure as a service

IaC. (1) infrastructure as code

IAP. (1) in-application programming

IBa. (1) issue benchmarks activity

IBD. (1) information-based domain

ICD. (1) initial capabilities document

ICE. (1) in-circuit emulator

ICOM. (1) input, control, output, and mechanism

ICOM code. (1) expression in one diagram that unambiguously identifies an arrow segment in another diagram. See also: common-environment coupling, content coupling, control coupling, data coupling, pathological coupling

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
ICOM label. (1) arrow label attached without a squiggle directly to the arrowhead of an output boundary arrow or to the arrowtail of an input, control, or mechanism boundary arrow (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.59) Note: An ICOM label associates a boundary arrow of a child diagram with an arrow label of an arrow attached to an ancestral box.

icon. (1) graphic displayed on the screen that represents a function of the computer system (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.22)


ICT. (1) information and communication technology (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

ICT product. (1) product which uses information and communication technologies (ICTs) and can be a part of an information system (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.8) Syn: information and communication technology product

ICT requirement. (1) requirement resulting from adoption of some information and communication technology (ICT) as a technical solution in the design process (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.7) Note: ICT technical solutions include web-based technologies and cloud servers. Syn: information and communication technology requirement

ICWG. (1) Interface Control Working Group (ISO/IEC/IEEE 16326:2019 Systems and software engineering -- Life cycle processes -- Project management, 5) Note: Depending on the size and complexity of a project, can be a group of people, a single person or a function

IDD. (1) interface design document (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation and Validation, 3.2)

IDE. (1) integrated development environment (ISO/IEC/IEEE 24765d:2015)

idea/mind mapping. (1) technique used to consolidate ideas created through individual brainstorming sessions into a single map to reflect commonality and differences in understanding, and to generate new ideas (ISO/IEC/IEEE 24765h:2019) Syn: idea mapping, mind mapping

ideal time. (1) a best-case estimate of the time needed for a developer or team to complete a task or deliver a feature (Software Extension to the PMBOK(R) Guide Fifth Edition)

IDEF. (1) integration definition for functional modeling (INCOSE Systems Engineering Handbook, 4th ed.)

IDEF0 model. (1) abstractly, a hierarchical set of IDEF0 diagrams that depict, for a specific purpose and from a specific viewpoint, the functions of a system or subject area, along with supporting glossary, text, and For Exposition Only (FEO) information (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.60) Note: Concretely, a set of model pages that include at least an A-0 context diagram and an A0 decomposition diagram, a glossary or specific glossary pages, one or more text pages to accompany each diagram, and FEO pages and model pages of other types as needed.

IDEF1X model. (1) set of one or more IDEF1X views, often represented as view diagrams that depict the underlying...
semantics of the views, along with definitions of the concepts used in the views (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.75)  

**identifier.** (1) unambiguous name, in a given naming context (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.2) (2) name, address, label, or distinguishing index of an object in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) within an IDEF0 model, a model name, a box name, or an arrow label (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.61)  

**identify risks.** (1) the process of identifying individual risks as well as sources of overall risk and documenting their characteristics (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)  

**identify stakeholders.** (1) the process of identifying the people, groups or organizations that could impact or be impacted by a decision, activity or outcome of the project, analyzing and documenting relevant information regarding their interests, involvement, interdependencies, influence, and potential impact on project success (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)  

**identifying relationship.** (1) specific (not many-to-many) relationship in which every attribute in the primary key of the parent entity is contained in the primary key of the child entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.79) See also: nonidentifying relationship [key style]  

**identity.** (1) inherent property of an instance that distinguishes it from all other instances (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.80) Note: Identity is intrinsic to the instance and independent of the instance’s property values or the classes to which the instance belongs.  


**IDL.** (1) Interface Definition Language (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)  

**idle.** (1) pertaining to a system or component that is operational and in service, but not in use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: busy, down, up  

**idle time.** (1) period of time during which a system or component is operational and in service, but not in use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: standby time See also: busy time, down time, set-up time, up time  


**IEEE.** (1) Institute of Electrical and Electronics Engineers (ISO/IEC/IEEE 24765:i:2020)  

**IES.** (1) integration enabling system (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.2)  

**IETF.** (1) Internet Engineering Task Force (ISO/IEC/IEEE 24765g:2018, 3.2)  

**if-then-else.** (1) single-entry, single-exit two-way branch that defines a condition, specifies the processing to be performed if the condition is met and, optionally, if it is not, and returns control in both instances to the statement
immediately following the overall construct (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

See also: case, jump, go to, dyadic selective construct, monadic selective construct


**iff. (1)** if and only if (ISO/IEC/IEEE 24765i:2020)

**IFPUG. (1)** International Function Point Users Group (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009) Note: membership governed, non-profit organization committed to promoting and supporting function point analysis and other software measurement techniques. The IFPUG maintains the definition of the direct descendent of the Albrecht 1984 FPA method


**IIOP-IOR. (1)** Internet Inter-ORB Protocol -- Interoperable Object Reference (ISO/IEC 14753:1999 Information technology -- Open Distributed Processing -- Interface references and binding, 4)

**ILF. (1)** internal logical file (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 4) (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

**illustration. (1)** graphic element set apart from the main body of text and normally cited within the main text (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.23)

(2) visually perceptible figure, artificially created to transmit specific information, excluding safety signs and graphical symbols (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.16) Note: used as the generic term for tables, figures, exhibits, screen captures, flow charts, diagrams, drawings, icons, and other graphic elements

**IM. (1)** incident manager (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.14)

**image processing. (1)** use of a data processing system to create, scan, analyze, enhance, interpret, or display images (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**immediate data. (1)** data contained in the address field of a computer instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: direct address, indirect address, n-level address, immediate instruction

**immediate instruction. (1)** computer instruction whose address fields contain the values of the operands rather than the operands' addresses (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: direct instruction, indirect instruction, absolute instruction, effective instruction, immediate data

**immunity. (1)** degree to which a product or system is resistant to attack (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6) See also: integrity

**immutable class. (1)** class for which the set of instances is fixed; its instances do not come and go over time (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.82) See also: mutable class, value class
IMP. (1) integrated master plan (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

impact analysis. (1) identification of all system and software products that a change request affects and development of an estimate of the resources needed to accomplish the change (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This includes determining the scope of the changes to plan and implement work, accurately estimating the resources needed to perform the work, and analyzing the requested changes’ cost and benefits.

imperative construct. (1) sequence of one or more steps not involving branching or iteration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


implementable standard. (1) template for a technology object (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 9.1.1)


implementation phase. (1) period of time in the software life cycle during which a software product is created from design documentation and debugged (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

implementation requirement. (1) requirement that specifies or constrains the coding or construction of a system or system component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: design requirement, functional requirement, interface requirement, performance requirement, physical requirement

implied addressing. (1) method of addressing in which the operation field of a computer instruction implies the address of the operands (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: direct address, indirect address, one-ahead addressing, relative address, repetitive addressing

implied needs. (1) needs that have not been stated but are actual needs (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.12) Note: Some implied needs only become evident when the system or software product is used in particular conditions.

import process. (1) process of incorporating the content of a transfer file into a target environment (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)
importer. (1) agent of the import process (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1)

imposed date. (1) a fixed date imposed on a schedule activity or schedule milestone, usually in the form of a "start no earlier than" and "finish no later than" date. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) - Sixth Edition)

impossible zone. (1) in a range of estimates, the region that is impossible under any circumstances to achieve (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For example, it is impossible to drive a car 500 miles in less than one hour, so the one-hour outcome for a 500-mile car trip is in the impossible zone for the estimate of how long it will take to drive 500 miles.

improvability. (1) inherent ability of an organization to support continual process improvement (ISO/IEC TR 33014:2013 Information technology--Process assessment--Guide for process improvement, 3.3)

IMS. (1) integrated master schedule (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

IN. (1) service incident process (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 4.1)

in-application programming (IAP). (1) capability of a microcontroller unit to fetch new program code and reprogram itself while the system is operating (ISO/IEC/IEEE 24765:2015) See also: in-system programming

in-circuit emulator (ICE). (1) hardware device used to debug the software of an embedded system (ISO/IEC/IEEE 24765:2015)

in-system programming (ISP). (1) capability of a microcontroller unit to allow the user to download new code (reprogram the unit), activated by restarting the unit (ISO/IEC/IEEE 24765:2015) See also: in-application programming

incentive fee. (1) a set of financial incentives related to cost, schedule, or technical performance of the seller (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


incident manager (IM). (1) role that has authority over all incidents and manages incident-related tasks (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.14) Note: This role may be combined with other roles. This role is a direct report (or shared role) with the Service Manager. The person can also be responsible for a Service Desk, if one exists.

incipient failure. (1) failure that is about to occur \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\)

include. (1) in UML, a relationship from a base use case to an included use case specifying how the behavior defined for the included use case can be inserted into the behavior defined for the base use case \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) \(2\) \([\text{information}]\) has either the information or a reference to the information \((ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 3.1.12)\)

income function. (1) objective function that characterizes the income generated by different values of the decision variable \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) See also: cost function

increment. (1) tested, deliverable version of a software product that provides new or modified capabilities \((Software Extension to the PMBOK(R) Guide Fifth Edition)\)

incremental analysis. (1) consideration of the relative differences between alternatives \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) Note: If the incremental benefit of a second alternative over the first is more than the incremental investment between them, the second alternative is a better investment than the first.

incremental benefit. (1) additional income from one alternative compared to another \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) Note: If Alternative A generates $10,000 and Alternative B generates $12,000, the incremental benefit between A and B is $2000.

incremental compiler. (1) compiler that completes as much of the translation of each source statement as possible during the input or scanning of the source statement \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) Note: Typically used for online computer program development and checkout. Syn: conversational compiler, interactive compiler, online compiler

incremental development. (1) software development technique in which requirements definition, design, implementation, and testing occur in an overlapping, iterative (rather than sequential) manner, resulting in incremental completion of the overall software product \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) See also: waterfall model, data structure-centered design, input-process-output, modular decomposition, object-oriented design

incremental investment. (1) avoidable additional investment between one alternative and another \((ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)\) Note: If Alternative A costs $10,000 and Alternative B costs $12,000, the incremental investment between A and B is $2000.

incremental life cycle. (1) an adaptive project life cycle in which the deliverable is produced through a series of iterations that successively add functionality within a predetermined time frame. The deliverable contains the necessary and sufficient capability to be considered complete only after the final iteration. \((A \text{ Guide to the Project Management Body})\)
incremental productivity. (1) productivity computed periodically during development (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

independence. (1) of software quality assurance (SQA), situation in which SQA is free from technical, managerial, and financial influences, intentional or unintentional (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2)

independent. (1) performed by an organization free from control by the supplier, developer, operator, or maintainer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

independent entity. (1) entity for which each instance can be uniquely identified without determining its relationship to another entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.84) Syn: identifier-independent entity See also: dependent entity [key style]

independent estimates. (1) a process of using a third party to obtain and analyze information to support prediction of cost, schedule or other items (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

independent evaluator. (1) individual or organization that performs an evaluation independently from developers and acquirers (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.30) Note: The individual or organization acting as developer or acquirer for the target system to be evaluated cannot become the independent evaluator for the system. The independent evaluator can be an organization.

independent state class. (1) state class that is not a dependent state class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.85) See also: dependent state class

independent verification and validation (IV&V). (1) verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)

index. (1) composite set of measures that reflect a concept (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)

indexed address. (1) address that must be added to the contents of an index register to obtain the address of the storage location to be accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: offset (2), relative address, self-relative address

indicative function point analysis. (1) an indication denoting the estimated size of an application or project, based exclusively on a conceptual data model or a data model in the third normal form (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)

indicator. (1) measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.13) (ISO/IEC/IEEE 15939:2017 Systems and...
software engineering--Measurement process, 3.10) (2) device or variable that can be set to a prescribed state based on the results of a process or the occurrence of a specified condition (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) measure of a discrete element of a domain (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)

indicator value. (1) numerical or categorical result assigned to an indicator (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.11)

indigenous error. (1) computer program error that has not been purposely inserted as part of an error-seeding process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

indirect address. (1) address that identifies the storage location of another address (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: The designated storage location can contain the address of the desired operand or another indirect address; the chain of addresses eventually leads to the operand. Syn: multilevel address See also: direct address, immediate data, indirect instruction, n-level address

indirect instruction. (1) computer instruction that contains indirect addresses for its operands (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: direct instruction, immediate instruction, absolute instruction, effective instruction

indirect labor. (1) human effort that is not directly associated with the units being produced (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: direct labor

indirect measure. (1) measure of an attribute that is derived from measures of one or more other attributes (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: An external measure of an attribute of a computing system (such as the response time to user input) is an indirect measure of attributes of the software as the measure will be influenced by attributes of the computing environment as well as attributes of the software.

indirect user. (1) person who receives output from a system, but does not interact with the system (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)---System and software quality models, 4.3.6) See also: direct user, secondary user

inductive assertion method. (1) a proof of correctness technique in which assertions are written describing program inputs, outputs, and intermediate conditions, a set of theorems is developed relating satisfaction of the input assertions to satisfaction of the output assertions, and the theorems are proved or disproved using proof by induction (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual)

infant mortality. (1) set of failures that occur during the early-failure period of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

influence diagram. (1) a graphical representation of situations showing causal influences, time ordering of events, and other relationships among variables and outcomes (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

influencer. (1) persons or groups that are not directly related to the acquisition or use of the product, but, who can affect the course of the project, positively or negatively, due to their position in the customer organization (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: stakeholder

informal group review. (1) informal review performed by three or more persons (ISO/IEC 20246:2017 Software and Systems Engineering Vocabulary)
informal review. (1) form of review that does not follow a defined process and has no formal documented output (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.6)

informal testing. (1) testing conducted in accordance with test plans and procedures that have not been reviewed and approved by a customer, user, or designated level of management (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: formal testing

information. (1) knowledge that is exchangeable amongst users, about things, facts, concepts, and so on, in a universe of discourse (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.6) (2) In information processing, knowledge concerning objects, such as facts, events, things, processes, or ideas, including concepts, that within a certain context has a particular meaning (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) meaningful data (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.23) (4) organized or structured data, processed for a specific purpose to make it meaningful, valuable, and useful in specific contexts. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: Although information will necessarily have a representation form to make it communicable, it is the interpretation of this representation (the meaning) that is relevant in the first place. See also: data

information analysis. (1) systematic investigation of information and its flow in a real or planned system (ISO/IEC 2382:2015 Information technology -- Vocabulary)

information architect. (1) person who develops the structure of an information space and the semantics for accessing required task objects, system objects, and other information (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.20)

information architecture. (1) &lt;human-centered&gt; structure of an information space and the semantics for accessing required task objects, system objects and other information (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.8) (2) structure of an information space and the semantics for accessing required task objects, system objects and other information (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.16) Note: The appropriate combination of organization, labeling, navigation schemes and retrieval mechanisms within an information space will facilitate task completion and efficient access to content.

information content. (1) set of metamodel and model instances found in a CDIF transfer (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

information development. (1) process of development concerned with determining what content and visuals shall be provided in product documentation and what the nature of the information shall be (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.21)

information development lead. (1) person who leads the activities of preparing documentation (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.23)

(2) person who leads the activities of preparing information for users (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.9)

information for use. (1) information provided by the supplier that provides the target audience with concepts, procedures and reference material for the safe, effective, and efficient use of a supported product during its life cycle (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.17) Note: Excludes supplementary information, which s outside the scope of information for use

Syn: instructions for use

information hiding. (1) software development technique in which each module's interfaces reveal as little as possible about the module's inner workings and other modules are prevented from using information about the module that is not in the module's interface specification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) containment of a design or implementation decision in a single module so that the decision is hidden from other modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: encapsulation


Note: The term information product is often used as a synonym.

An information item can be produced in several versions during a system, software, or service life cycle. See also: document, information product

information item content. (1) information included in an information item, associated with a system, product or service, to satisfy a requirement or need (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.13)

information item type. (1) group of information items consistent with a pre-arranged set of generic criteria (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.14) Syn: generic document type

information management. (1) in an information processing system, the functions of controlling the acquisition, analysis, retention, retrieval, and distribution of information (ISO/IEC 2382:2015 Information technology -- Vocabulary)
information management system. (1) facilities, processes and procedures used to collect, store and distribute information between producers and consumers of information in physical or electronic format (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


information processing. (1) systematic performance of operations upon information, which includes data processing and can include operations such as data communication and office automation (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: The term information processing is not a synonym for data processing

information processing requirements. (1) the set of functions required by the commissioning user of the application software product (excluding any technical and quality requirements) (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 1.1) See also: software

information processing system. (1) one or more data processing systems and devices, such as office and communication equipment, that perform information processing (ISO/IEC 2382:2015 Information technology -- Vocabulary)

information product. (1) one or more indicators and their associated interpretations that address an information need (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.13) Note: Often used as a synonym of information item. See also: information item

information provisioning. (1) collection of all the infrastructure tools, software applications, non-automated elements, data sets, user documentation, and organizational structures which serve to supply information to the business (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.20)

information quality. (1) degree to which an information product satisfies the stated and implied needs when used under specified conditions (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.20)

information radiator. (1) a large and frequently updated display of project information that is continually visible to the project team and other stakeholders (Software Extension to the PMBOK(R) Guide Fifth Edition)

information retrieval (IR). (1) actions, methods, and procedures for obtaining information on a given subject from stored data (ISO/IEC 2382:2015 Information technology -- Vocabulary)

information security. (1) preservation of confidentiality, integrity and accessibility of information (ISO/IEC/IEEE 24765c:2014) Note: In addition, other properties such as authenticity, accountability, non-repudiation and reliability can also be involved.

information security incident. (1) single or a series of unwanted or unexpected information security events that have a significant probability of compromising business operations and threatening information security (ISO/IEC/IEEE 24765c:2014) Note: [ISO/IEC 27000:2009]
information security policy. (1) document that states, in writing, how an organization plans to protect its physical and information technology assets (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.15)

information structure. (1) structure that provides information about an IT asset in order to facilitate its management (ISO/IEC 19770-4:2017 Information technology -- IT asset management -- Part 4: Resource utilization measurement, 3.13) Syn: info struct

information structure creator. (1) entity that initially creates an information structure (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.19) Syn: info struct creator

information structure id. (1) globally unique value for every information structure created (ISO/IEC 19770-1:2017 Information technology -- IT asset management systems--Requirements, 3.20) Syn: info struct ID

information system. (1) information processing system, together with associated organizational resources such as human, technical, and financial resources, which provides and distributes information (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) all of the functions (input, output, transport, processing, and storage) of an application, databases, technical facilities, and manual procedures which support business processes (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.21) (3) one or more computer systems and communication systems together with associated organizational resources such as human, technical, and financial resources that provide and distribute information (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.24) (4) system that is comprised of software, hardware, communication facility, data, and the people who use it in a given environment to satisfy their information processing needs (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.10) See also: application

information technology. (1) resources required to acquire, process, store and disseminate information (ISO/IEC/IEEE 24765:2014) (2) development, maintenance, and use of technology to acquire, process, store and distribute digital information (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.24) Note: includes Communication Technology (CT) and the composite term Information and Communication Technology (ICT) Syn: IT

information technology project. (1) temporary endeavor undertaken to create or change a unique information technology product, system, or service (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.7) Syn: IT project, information technology (IT) project

information technology service. (1) service that makes use of IT systems as tools to provide value to an individual user or a business by facilitating results the user or business wants to achieve (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.2) Syn: IT service

information type. (1) class of topics that addresses a particular user question (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation,
4.23) (2) category of topics, such as concepts, tasks, or reference (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.18)

information viewpoint. (1) viewpoint on an ODP system and its environment that focuses on the semantics of information and information processing (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.1.1.2)

Information-based domain (IBD). (1) realm of activity for which information is the most valuable asset (ISO/IEC 24744:2014 Software Engineering--Metamodelfor development methodologies, 3.1) Note: Information creation, manipulation, and dissemination are the most important activities within information-based domains. Typical information-based domains are software and systems engineering, business process reengineering, and knowledge management

infrastructure. (1) hardware and software environment to support computer system and software design, development, and modification (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.25) (2) facilities such as power, cooling, and physical security of the data center, networking, hardware, and software needed to support the systems life cycle and maintain information technology (IT) services (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: Does not include the associated people or processes. In DevOps, software-defined infrastructure enables elasticity. Syn: ecosystem See also: IT infrastructure

infrastructure as code (IaC). (1) definition, management, and provision of infrastructure components using software (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: In DevOps, infrastructure as code facilitates the automation of the systems life cycle, enabling consistency, performance, and security across the system and resources

inheritance. (1) a semantic notion by which the responsibilities (properties and constraints) of a subclass are considered to include the responsibilities of a superclass, in addition to its own, specifically declared responsibilities (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.86)

inherited attribute. (1) attribute that is a characteristic of a class by virtue of being an attribute of a generic ancestor (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.87) (2) attribute that is a characteristic of a category entity by virtue of being an attribute in its generic entity or a generic ancestor entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.87)

inherited error. (1) error carried forward from a previous step in a sequential process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

initial Ent. (1) Ent that is referenced by later Ents (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.14) Note: The initial Ent is typically a record of the first transaction between software licensor and end customer. An initial Ent is a type of primary Ent. Syn: initial entitlement schema

initial function point analysis. (1) a function point analysis performed at the beginning of a project (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)
initial investment. (1) investment required just to start an activity (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: first cost

initial marking (of the net). (1) set of initial place markings given with the high-level net definition (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.10)

initial marking of a place. (1) special marking of a place, defined with the net (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.11) Syn: initial marking of place

initial program loader. (1) bootstrap loader used to load that part of an operating system needed to load the remainder of the operating system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

initial risk. (1) estimated risk before applying risk reduction measures (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.6)

initialization section. (1) optional list of unconditional actions to be executed sequentially before the first condition is examined (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.13) Note: It can be written in the row which follows that of the table heading.

initialize. (1) to set a variable, register, or other storage location to a starting value (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: clear, reset

initiating object. (1) object causing a communication (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.4.1)

initiating process group. (1) those processes performed to define a new project or a new phase of an existing project by obtaining authorization to start the project or phase (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

initiation trigger. (1) control information or functional control data that is responsible for activating the intended transaction (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B)

initiative. (1) degree to which the IT service recognizes users? goals and service suggests changes to meet users? needs (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.7.2)

initiator. (1) person or organization that has both the ability and authority to start a project (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

injection slot. (1) point where the recoverability of the system under test (SUT) is tested by injecting a disturbance while a workload is being run (ISO/IEC 25045:2010 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation module for recoverability, 4.3)

inline code. (1) sequence of computer instructions that is physically contiguous with the instructions that logically precede and follow it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

inner cardinality. (1) number of allowed instances of the relationship from the viewpoint of a single instance of the data object planning a role (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data


input argument. (1) designation given to an operation argument that will always have a value at the invocation of the operation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: output argument

input arrow. (1) arrow or arrow segment that expresses IDEF0 input (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.63) Note: That is, an object type set whose instances are transformed by a function into output. The arrowhead of an input arrow is attached to the left side of a box.

input assertion. (1) logical expression specifying one or more conditions that program inputs must satisfy in order to be valid (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: loop assertion, output assertion, inductive assertion, method

input loopback. (1) loopback of output from one function to be input for another function in the same diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.64)

input place (of a transition). (1) place connected to the transition by an input arc (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.13)

input primitive. (1) the effort to develop software products, expressed in units of staff-hours (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

input routine. (1) those activities required to obtain the logical record to be processed next (ISO/IEC/IEEE 24765a:2011) Note: If there are no more logical records to be processed, the end-of-input condition becomes true.

input-process-output. (1) software design technique that consists of identifying the steps involved in each process to be performed and identifying the inputs to and outputs from each step (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A refinement called hierarchical input-process-output identifies the steps, inputs, and outputs at both general and detailed levels of detail See also: data structure-centered design, input-process-output chart, modular decomposition, object-oriented design, rapid prototyping
input-process-output (IPO) chart. (1) A diagram of a software system or module, consisting of a rectangle on the left listing inputs, a rectangle in the center listing processing steps, a rectangle on the right listing outputs, and arrows connecting inputs to processing steps and processing steps to outputs. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: summarizes the process activities and their inputs and outputs from/to external actors; some inputs are categorized as controls and enablers. Syn: IPO diagram See also: block diagram, box diagram, bubble chart, flowchart, graph, structure chart

inspection. (1) Static analysis technique that relies on visual examination of development products to detect errors, violations of development standards, and other problems. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) Examination of a work product to determine whether it conforms to documented standards. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) Examining or measuring to verify whether an activity, component, product, result, or service conforms to specified requirements. (ISO/IEC/IEEE 24765h:2019) (4) Formal review of a work product to identify issues, which uses defined team roles and measurement to improve the review process. (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.8) Note: Inspections are peer examinations led by impartial facilitators who are trained in inspection techniques. Determination of remedial or investigative action for an anomaly is a mandatory element of a software inspection, although the solution could be determined outside the inspection meeting. Types include code inspection and design inspection. See also: static testing

inspection-based evaluation. (1) Evaluation based on the judgment of one or more evaluators who examine or use a system to identify potential usability problems, including deviations from established criteria. (ISO/IEC 25066:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for Usability--Evaluation Report, 3.10) Note: The evaluators making the inspections typically are usability specialists, but can also include end users and members of the design team. Inspection-based evaluation can be conducted by machines in some cases, e.g., when consistency with required terminology is being evaluated. Established criteria typically include user requirements, usability guidelines in standards, design conventions contained in manufacturer guidelines and style guides, task models to be supported, as well as standardized principles.

installability. (1) Degree of effectiveness and efficiency with which a product or system can be successfully installed or uninstalled in a specified environment. (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.8.2)

installability testing. (1) Type of portability testing conducted to evaluate whether a test item or set of test items can be installed as required in all specified environments. (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.19)

installation and checkout phase. (1) 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. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

installation manual. (1) Document that provides the information necessary to install a system or component, set initial parameters, and prepare the system or component for operational use. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: diagnostic manual, operator manual, programmer manual, support manual,

---


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
installed function point count. (1) an application function point count related to a set of installed systems

instance. (1) discrete, bounded thing with an intrinsic, immutable, and unique identity (IEEE 1320.2-1998 (R2004)
IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.89) (2)

instance of benchmarking. (1) set of operations, described specifically, used in the execution of a particular benchmarking according to a given method (ISO/IEC 29155-1:2017 Systems and software engineering—Information technology project performance benchmarking framework—Part 1: Concepts and definitions, 2.6)

instance-level attribute. (1) mapping from the instances of a class to the instances of a value class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.90)

instance-level operation. (1) mapping from the (cross product of the) instances of the class and the instances of the input argument types to the (cross product of the) instances of the other (output) argument types (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.91)

instance-level responsibility. (1) responsibility that applies to each instance of the class individually (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.92) See also: class-level responsibility

instantiated trace link. (1) trace link derived by applying binding in a member product (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 3.2)

instantiation. (1) process of substituting specific data, instructions, or both into a generic program unit to make it usable in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) of an &lt;X&gt; template, an &lt;X&gt; produced from a given &lt;X&gt; template and other necessary information (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.16) (3)

institutional knowledge. (1) knowledge from accepted sources, including standards, academic sources, domain and industry bodies of knowledge and organizational knowledge (ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.4)

institutionalization. (1) ingrainned way of doing business that an organization follows routinely as part of its corporate activities
culture (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

instruction counter. (1) register that indicates the location of the next computer instruction to be executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: program counter

instruction cycle. (1) process of fetching a computer instruction from memory and executing it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: instruction time

instruction format. (1) number and arrangement of fields in a computer instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: address field, address format, operation field

instruction length. (1) number of words, bytes, or bits needed to store a computer instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: instruction format

instruction modifier. (1) word or part of a word used to alter a computer instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

instruction set. (1) complete set of instructions recognized by a given computer or provided by a given programming language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: instruction repertoire

instruction time. (1) time it takes a computer to fetch an instruction from memory and execute it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: instruction cycle

instructional mode. (1) usage mode that is intended to teach the use of software in performing tasks (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.25)

instrument. (1) in software and system testing, to install or insert devices or instructions into hardware or software to monitor the operation of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

instrumentation. (1) devices or instructions installed or inserted into hardware or software to monitor the operation of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

integer type. (1) data type whose members can assume only integer values and can be operated on only by integer arithmetic operations, such as addition, subtraction, and multiplication (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: character type, enumeration type, logical type, real type

integrate. (1) to combine software components, hardware components, or both into an overall system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to pull in the changes from one child branch into its parent (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


integrated development environment (IDE). (1) set of software tools or applications to provide comprehensive facilities for software development (ISO/IEC/IEEE 24765d:2015)

integrated repository. (1) planned and controlled storage of information pertinent to the systems engineering effort (ISO/IEC/IEEE 24748-4:2016 Systems and software engineering-Life cycle management-Part 4: Systems engineering planning, 4.6) Note: The integrated repository typically includes key data, e.g., schema, models, tools, technical management decisions, process analysis information, requirement changes, process and product metrics, trade-offs and other analyses.
integrated team. (1) group of people with complementary skills and expertise who are committed to delivering specified work products in timely collaboration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Integrated team members provide skills and advocacy appropriate to all phases of the work products’ life and are collectively responsible for delivering work products as specified. An integrated team includes empowered representatives from organizations, disciplines, and functions that have a stake in the success of the work products.

integration. (1) process of combining software components, hardware components, or both into an overall system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) activities of combining several implemented system elements and activating the interfaces to form a realized system (product or service) that enables interoperation between the system elements and with other systems to satisfy system requirements, architecture characteristics and design properties (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.5) Note: Integration can apply to the implemented system elements which compose a system and the necessary life-cycle related activities, and also to connect a system-of-interest with external interoperating systems or enabling systems.

integration definition for functional modeling (IDEF). (1) family of modeling languages in the fields of systems and software engineering that provide a multiple-page (view) model of a system that depicts functions and information or product flow (INCOSE Systems Engineering Handbook, 4th ed.) Note: Boxes illustrate functions and arrows illustrate information and product flow. Alphanumeric coding is used to denote the view.

integration engineering. (1) set of activities that defines, analyzes and executes integration across the life cycle, including interactions with other life cycle processes (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.6)

integration management. (1) set of activities that plans, assesses, and controls the integration activities and all related activities (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.7)

integration test. (1) progressive linking and testing of programs or modules in order to ensure their proper functioning in the complete system (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: integration testing

integration testing. (1) testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

integrity. (1) degree to which a system, product, or component prevents unauthorized access to, or modification of, computer programs or data (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6.2) (2) degree to which an IT service prevents unauthorized access to or modification of data, whether accidentally or intentionally (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.3.2) See also: immunity

integrity assurance authority. (1) independent person or organization responsible for certifying compliance with the integrity-level requirements (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.5.4)
**integrity level.** (1) value representing project-unique characteristic, such as complexity, criticality, risk, safety level, security level, desired performance, and reliability, that define the importance of the system, software, or hardware to the user. (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.15) (2) degree to which software complies or must comply with a set of stakeholder-selected software or software-based system characteristics defined to reflect the importance of the software to its stakeholders. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) symbolic value representing a degree of compliance within an integrity level scheme. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) claim of a system, product, or element that includes limitations on a property's values, the claim's scope of applicability, and the allowable uncertainty regarding the claim's achievement. (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.3.1) (5) required degree of confidence that the system-of-interest meets the associated integrity level claim. (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.7) Note: Generally, the intention is that maintaining limitations on a property's values related to the relevant items will result in maintaining system risks within limits. The words 'integrity level' form an indivisible label and do not depend on a concept of integrity by itself. An integrity level is different from the likelihood that the integrity level claim is met but they are closely related. The word 'confidence' implies that the definition of integrity levels can be a subjective concept. integrity levels are defined in terms of risk and hence, cover safety, security, financial and any other dimension of risk that is relevant to the system-of-interest.

**integrity level assurance authority.** (1) independent person or organization responsible for certifying compliance with the integrity level requirements. (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.8)

**integrity level claim.** (1) claim representing a requirement for a risk reduction measure identified in the risk treatment process of the system-of-interest. (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.10)

**integrity level definition authority.** (1) person or organization responsible for defining integrity levels and integrity level requirements. (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.9)

**integrity level requirement.** (1) set of requirements that, when met, will provide a level of confidence in the associated integrity level claim commensurate with the associated integrity level. (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.3.2) (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.11) (2) set of requirements that, when met, will provide a level of confidence in the associated integrity level claim (3.3.4) commensurate with the associated integrity level (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.3.2)

**integrity level requirements.** (1) set of specified requirements imposed on aspects related to a system, product, or element and associated activities in order to show the achievement of the assigned integrity level (that is, meeting its claim) within the required limitations on uncertainty; this includes the evidence to be obtained. (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.3.2)
integrity level scheme. (1) set of system characteristics (such as complexity, risk, safety level, security level, desired performance, reliability, or cost) selected as important to stakeholders, and arranged into discrete levels of performance or compliance (integrity levels), to help define the level of quality control to be applied in developing or delivering the software (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

intellectual property. (1) output of creative human thought process that has some intellectual or informational value (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.19) Note: Intellectual property can be protected by patents, copyrights, trademarks, or trade secrets.

intended use. (1) exhaustive range of functions or foreseen applications defined and designed by the supplier of the product (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.21) Note: Functions or applications not listed by the supplier are excluded from the intended use of the product. Additional or modified functions or applications resulting from modifications not sanctioned by the supplier of the product are excluded from the intended use.

inter-integrated circuit bus (I2C). (1) bi-directional two-wire serial bus that provides a communication link between integrated circuits (ISO/IEC/IEEE 24765d:2015)

interaction. (1) action that takes place with the participation of the environment of the object (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.3) Note: The identity is expressed in relevant phenomenological terms. Generally, an interaction identity can be categorized as energy transfer, matter transfer, or information transfer.

interaction group. (1) subset of the objects participating in a binding managed by the group function (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.4.1.1)

interaction point. (1) location at which there exists a set of interfaces (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.12)

interactive. (1) pertaining to a system or mode of operation in which each user entry causes a response from or action by the system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) when the user communicates with the computer in a conversational-type manner (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) See also: batch, conversational, online, real-time

interactive language. (1) nonprocedural language in which a program is created as a result of interactive dialog between the user and the computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The system provides questions, forms, and so on, to aid the user in expressing the results to be achieved. See also: declarative language, rule-based language

interactive system. (1) combination of hardware, software and/or services that receives input from and communicates output to users (ISO/IEC 25063:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description) Note: This includes, where appropriate, packaging, branding, user documentation, online help, support and training.

interchange reference point. (1) reference point at which an external physical storage medium can be introduced into the system (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model:
interconnection feature. (1) property by which members of an interconnection perspective are characterized (IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection--Characterization of Interconnections, 3.6)

interface. (1) shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical signal exchanges, and other characteristics (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) hardware or software component that connects two or more other components for the purpose of passing information from one to the other (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to connect two or more components for the purpose of passing information from one to the other (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) to serve as a connecting or connected component as in (2) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (5) declaration of the meaning and the signature for a property or constraint (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.93) (6) shared boundary across which information is passed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (7) task's external specification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (8) abstraction of the behavior of an object that consists of a subset of the interactions of that object together with a set of constraints on when they can occur (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.5) (9) description of a set of possible operations that a client is allowed to request of an object (ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3.5) (10) named set of operations that characterize the behavior of an entity (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (11) set of logical or physical characteristics required to exist at a common boundary or connection between system elements (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.8)

interface control. (1) in configuration management, the administrative and technical procedures and documentation necessary to identify functional and physical characteristics between and within configuration items provided by different developers, and to resolve problems concerning the specified interfaces (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in configuration management, the process of identifying all functional and physical characteristics relevant to the interfacing of two or more configuration items provided by one or more organizations and ensuring that proposed changes to these characteristics are evaluated and approved prior to implementation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control

interface design document (IDD). (1) documentation that describes the architecture and design interfaces between system and components (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.16) (2) description of the architecture and design of interfaces between system and components (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: These descriptions include control algorithms, protocols, data contents and formats, and performance. See also: interface requirements specification (IRS)

interface requirement. (1) requirement that specifies an external item with which a system or system component must interact, or that sets forth constraints on formats, timing, or other factors caused by such an interaction
interface requirements specification (IRS). (1) documentation that specifies requirements for interfaces between or among systems and components (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.17) Note: These requirements include constraints on formats and timing. See also: interface specification, interface design document

interface role. (1) role of a community, identifying behavior which takes place with the participation of objects that are not members of that community (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.5)


interface specification. (1) description of essential functional, performance, and design requirements and constraints at a common boundary between two or more system elements (ISO/IEC/IEEE 24765:2016) (2) document that specifies the interface characteristics of an existing or planned system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This includes interfaces between humans and hardware or software, as well as interfaces between humans themselves. See also: interface requirements specification

interface task. (1) task that is part of the application, which interfaces to the external environment (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

interface testing. (1) testing conducted to evaluate whether systems or components pass data and control correctly to one another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: component testing, integration testing, system testing, unit test


interim function point analysis. (1) an analysis to determine the size of an interim change of requirements during a new development project or an enhancement project, which is an analysis to determine the scope of an addition, a change, or a deletion of functional specifications (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Note: Both the change in the application functional size and the project functional size can be the subject of this analysis.

interleave. (1) to alternate the elements of one sequence with the elements of one or more other sequences so that each sequence retains its identity (ISO/IEC/IEEE 12456:2017 Systems and software engineering-Vocabulary)

intermediate product. (1) system or software product of the development process that is used as inputs to other stages of the development process (ISO/IEC 25041: 2012 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation guide for developers, acquirers and independent evaluators, 4.10) See also: intermediate system or software product

intermediate profile. (1) profile targeted at very small entities (VSEs) involved in the development of more than one project in parallel with more than one work team (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for
**Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.30**

**intermediate software product needs. (1)** needs that can be specified as quality requirements by internal measures (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**intermediate system or software product. (1)** product of the system or software development process that is used as input to another stage of its development process (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.15) Syn: intermediate software product, intermediate system product See also: intermediate product

**intermittent fault. (1)** temporary or unpredictable fault in a component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: random failure, transient error

**internal action. (1)** action which takes place without the participation of the environment of the object (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.3)

**internal arrow. (1)** arrow connected at both ends (source and use) to a box in a diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.65) See also: boundary arrow

**internal attribute. (1)** measurable property of an entity which can be derived purely in terms of the entity itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Internal attributes are those that relate to the internal organization of the software and its development.

**internal event. (1)** means of synchronization between two tasks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**internal logical file (ILF). (1)** user-recognizable group of logically related data or control information maintained within the boundary of the application being measured (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.39) (2) a logical group of permanent data seen from the perspective of the user that an application uses and maintains (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted. An internal logical file is a type of base functional component. See also: external interface file

**internal measure. (1)** measure of the product itself, either direct or indirect (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The number of lines of code, complexity measures, the number of faults found in a walk through and the Fog Index are all internal measures made on the product itself.

**internal measure of software quality. (1)** measure of the degree to which a set of static attributes of a software product satisfies stated and implied needs for the software product to be used under specified conditions (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.16) (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.7) Note: Static attributes include those that relate to the software architecture, structure and its components. Static attributes can be verified by review, inspection, simulation, or automated tools. See also: external measure of software quality

**internal quality. (1)** totality of attributes of a product that determine its ability to satisfy stated and implied needs when
internal service provider. (1) organization providing services internal to the very small enterprise (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.16)

internal task-structuring criteria. (1) category of the task-structuring criteria addressing how internal objects are mapped to internal tasks and when an internal task is activated (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

internal variability. (1) variability defined from an engineer's perspective and not visible to customers (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.6)

internal standard (IS). (1) standard that is adopted by an international standardizing/standards organization and made available to the public (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.11) Note: [ISO/IEC Directives, Part 2]

duration. (1) period of time the project requires or is expected to require (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

duration. (1) amount of time required to complete a project activity or work package (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

interpret. (1) to translate and execute each statement or construct of a computer program before translating and
executing the next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assemble, compile

**interpretability. (1)** level of understanding how the underlying technology works (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.42)

**interpreter. (1)** computer program that translates and executes each statement or construct of a computer program before translating and executing the next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembler, compiler

**interpretable code. (1)** computer instructions and data definitions expressed in a form that can be recognized and processed by an interpreter (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembly code, compiler code, machine code


**interrogation. (1)** interaction consisting of one interaction, the invocation, initiated by a client object, resulting in the conveyance of information from that client object to a server object, and requesting a function to be performed by the server object, followed by a second interaction, the termination, initiated by the server object, resulting in the conveyance of information from the server object to the client object in response to the invocation (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.4) Note: In interrogations, invocations and terminations are always paired. Announcements do not have terminations. Thus there is no possibility of an operation consisting of an invocation followed by a sequence of associated terminations.


**interrupt controller. (1)** functional unit (integrated circuit) that determines the source and priority of interrupt requests and manages their execution (ISO/IEC/IEEE 24765d:2015)

**interrupt latency. (1)** delay between a computer system's receipt of an interrupt request and its handling of the request (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: interrupt priority

**interrupt mask. (1)** mask used to enable or disable interrupts by retaining or suppressing bits that represent interrupt requests (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**interrupt priority. (1)** importance assigned to a given interrupt request (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This importance determines whether the request will cause suspension of the current process and, if there are several outstanding interrupt requests, which will be handled first.

**interrupt request. (1)** signal or other input requesting that the currently executing process be suspended to permit performance of another process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
interrupt service routine. (1) routine that responds to interrupt requests by storing the contents of critical registers, performing the processing required by the interrupt request, restoring the register contents, and restarting the interrupted process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: ISR

interval scale. (1) scale in which the measurement values have equal distances corresponding to equal quantities of the attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: ordinal scale, nominal scale, ratio scale

interviews. (1) a formal or informal approach to elicit information from stakeholders by talking to them directly (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

interworking reference point. (1) reference point at which an interface can be established to allow communication between two or more systems (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 15.3.3)

Intranet. (1) managed network operating within an organization with controlled and limited access (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.14) Note: More than one connected or isolated intranet can exist within an organization. Syn: intranet

intrinsic. (1) specification that a property is total (i.e., mandatory), single-valued, and constant (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.94)

intrinsic relationship. (1) relationship that is total, single-valued, and constant from the perspective of (at least) one of the participating classes, referred to as a dependent class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.95) Note: Such a relationship is considered to be an integral part of the essence of the dependent class. See also: nonintrinsic relationship


invariant. (1) assertion that is always be true for a specified segment or at a specified point of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) predicate that a specification requires to be true for the entire life time of a set of objects (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.28)

invariant schema. (1) set of predicates on one or more information objects which must always be true (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 6.1.1) Note: The predicates constrain the possible states and state changes of the objects to which they apply. Thus, an invariant schema is the specification of the types of one or more information objects that will always be satisfied by whatever behavior the objects might exhibit.

inverse engineering. (1) process of obtaining a high-level representation of the software from the source code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Inverse engineering provides a more abstract view of the system with the intent of recapturing design and requirements information. See also: reverse engineering

invitation for bid (IFB). (1) generally, this term is equivalent to request for proposal. However, in some application
areas, it may have a narrower or more specific meaning. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**invocation.** *(1)* mapping of a parallel initiation of activities of an integral activity group that perform a distinct function and return to the initiating activity *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)* See also: instance, iteration, mapping

**invocation deliver.** *(1)* signal in the implicitly defined signal interface of a server computational object which has the same name and parameters as the invocation of an interrogation or announcement in the original operation interface *(ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.8)*

**invocation submit.** *(1)* signal in the implicitly defined signal interface of a client computational object which has the same name and parameters as the invocation of an interrogation or announcement in the original operation interface *(ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.7)* See also: termination submit

**IOC.** *(1)* initial operational capability *(IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)*


**IOT&E.** *(1)* initial operational test and evaluation *(IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)*

**IP.** *(1)* Internet Protocol *(ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)*

**IPO chart.** *(1)* input-process-output chart *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**IPR.** *(1)* intellectual property rights *(ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)*

**IPSE.** *(1)* integrated programming support environment *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: programming support environment

**IPT.** *(1)* integrated product team *(IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)*

**IRR.** *(1)* integration readiness review *(IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)*

**irreducible.** *(1)* decision attribute (criterion) that cannot be expressed in terms of money *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**IRS.** *(1)* interface requirements specification *(IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)*

**IS.** *(1)* international standard *(ISO/IEC/IEEE 24765a:2011)*

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
ISO file. (1) file image of an entire CD or DVD that is encoded according to ISO 9660 (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)
isochronicity. (1) relation between adjacent pairs of actions in a sequence, in which every adjacent pair of actions occupies unique, equally-sized, adjacent intervals in time (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.3.2)
ISR. (1) interrupt service routine (ISO/IEC/IEEE 24765d:2015)
issue. (1) uniquely identifiable entry in an issue-tracking system that describes a problem or an enhancement (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) point or matter in question or in dispute, or a point or matter that is not settled and is under discussion or over which there are opposing views or disagreements (ISO/IEC 24765h:2019) (3) observation that deviates from expectations (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.9) (4) a current condition or situation that may have an impact on the project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: The record of an issue includes its identifier and brief description, and often identifies the environment associated with it, its status, severity, priority, and resolution, as well as dependencies, details on replicating or solving a problem, the persons associated with it, attachments, and its change history. See also: problem report
issue log. (1) project document used to monitor elements under discussion or in dispute between project stakeholders (ISO/IEC/IEEE 24765h:2019) (2) a project document where information about issues is recorded and monitored (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
IT. (1) Information Technology (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)
IT asset. (1) item, thing, or entity that can be used to acquire, process, store and distribute digital information and has potential or actual value to an organization (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.25) Note: Information per se, independent of IT hardware and software assets, can be considered an asset, but it is not considered an IT asset. The collective set of IT assets is also referred to as the IT infrastructure. Syn: information technology asset See also: digital asset
IT asset management (ITAM). (1) coordinated activity of an organization to realize value from IT assets (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.26) Syn: information technology asset management
IT asset management plan. (1) documented information that specifies the activities, resources and timescales required for an individual information technology (IT) asset, or a grouping of IT assets, to achieve the organization’s IT asset management objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.27) Syn: information technology asset management plan
IT asset management system (ITAMS). (1) management system for information technology (IT) asset management, whose function is to establish the IT asset management policy and IT asset management objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--
Requirements, 3.28) Syn: information technology asset management system

IT asset portfolio. (1) IT assets that are within the scope of the IT asset management system (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.29)

Note: A portfolio is typically established and assigned for managerial control purposes. Portfolios for IT hardware might be defined by category (e.g. servers, PCs, mobile devices). Software portfolios might be defined by software publisher, or by platform (e.g. PC, server, mainframe). An IT asset management system can encompass multiple IT asset portfolios. Syn: information technology asset portfolio See also: asset portfolio

IT infrastructure. (1) all the technical components, system software, databases and data files and deployed application software, technical procedures, and technical documentation used to make the information available (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.22) (2) combined set of IT assets for developing, maintaining, and using IT services (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.30)

IT infrastructure management. (1) domain responsible for all of the tasks and activities aimed at managing, maintaining, and renewing the IT infrastructure of the information system, including the operation of the information system (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.23)

Note: IT infrastructure management includes all of the tasks, responsibilities and activities that aim for a correct technical operation of the information system, consisting of hardware, (system) software, and data sets.

The IT infrastructure management organization is responsible for running the application software in the production environment.

IT project dataset. (1) classified group of data records, into which collected data records are selected by pre-defined criteria (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 3.17) Syn: information technology project dataset

IT service adaptability. (1) degree to which an IT service can configure itself or be modified to meet new needs (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.7) See also: adaptability

IT service function. (1) collection of related steps performed as a part of an IT service, or features provided by an IT system (ISO/IEC TS 25025:2021, Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Measurement of IT service quality, 3.4)

IT service interface appearance. (1) degree to which the interface of the service has an appearance or other physical properties that are pleasing and satisfying for the user (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.5.3)

IT service maintainability. (1) degree of effectiveness and efficiency with which the IT service can be modified by the service provider (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.8)

IT service quality. (1) degree to which an IT service satisfies stated and implied needs when used under specified conditions (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation
IT service recoverability. (1) degree to which, in the event of an interruption or a failure or disaster, the original IT service and its functions and data can be re-established and made accessible. (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.4.2) See also: recoverability

IT service reliability. (1) degree to which an IT service provides consistent and stable IT service outcomes. (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.4) See also: reliability, software reliability

IT system. (1) system which uses information technologies (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.3) (2) set of one or more computers, associated software, peripherals, terminals, human operations, physical processes, information transfer means, that form an autonomous whole, capable of performing information processing and/or information transfer. (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.5)

ITAM. (1) information technology asset management (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.26) Syn.: IT asset management

ITAMS. (1) information technology asset management system (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.28) Syn.: IT asset management system

iteration. (1) process of performing a sequence of steps repeatedly (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) a single execution of the sequence of steps in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) short time frame in which a set of software features is developed, leading to a working product that can be demonstrated to stakeholders. (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.10) Note: One or more iterations comprise an instance. See also: instance, invocation, mapping

iterative development. (1) repeated use of concurrent planning, developing, and testing activities (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.11)

iterative life cycle. (1) a project life cycle where the project scope is generally determined early in the project lifecycle, but time and cost estimates are routinely modified as the project team understanding of the product increases. Iterations develop the product through a series of repeated cycles, while increments successively add to the functionality of the product. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: incremental life cycle

ITT. (1) invitation to tender (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 3.2)

IV&V. (1) independent verification and validation (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2) Syn: IV & V

IVV. (1) integration, verification, validation (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic
job. (1) user-defined unit of work that is to be accomplished by a computer (ISO/IEC 25023:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Measurement of system and software product quality, 4.3) See also: job control language, job step, job stream

job control language (JCL). (1) language used to identify a sequence of jobs, describe their requirements to an operating system, and control their execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

job function. (1) group of engineering processes that is identified as a unit for the purposes of work organization, assignment, or evaluation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

job step. (1) user-defined portion of a job, explicitly identified by a job control statement (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A job consists of one or more job steps.

job stream. (1) sequence of programs or jobs set up so that a computer can proceed from one to the next without the need for operator intervention (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: run stream

join. (1) junction at which an arrow segment (going from source to use) merges with one or more other arrow segments to form a root arrow segment (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.66) Note: can denote bundling of arrows, meaning the inclusion of multiple object types within an object type set

joining action. (1) action shared between two or more chains resulting in a single chain (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.3)

JPEG. (1) Joint Photographic Experts Group (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5) Note: image format

JPG. (1) Joint Photographic Group (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5) Note: image format

program statement that causes a departure as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) departure described in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: transfer
junction. (1) point at which either a root arrow segment divides into branching arrow segments or arrow segments join into a root arrow segment (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.67)
KDM. (1) knowledge discovery meta-model (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)
KDM entity. (1) meta-model element (as well as the corresponding model elements) that represents a thing of significance of the system of interest, about which information needs to be known or held (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)
KDM instance. (1) collection of KDM model elements that represent one or more views of the system of interest (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)
KDM model. (1) meta-model element (as well as the corresponding model elements) that is a container for a KDM view (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)
KDM relationship. (1) meta-model element (as well as the corresponding model elements) that represents some semantic association between elements of the system of interest (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)
kernel. (1) that portion of an operating system that is kept in main memory at all times (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) a software module that encapsulates an elementary function or functions of a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: resident control program See also: nucleus, supervisory program
kernel entity. (1) a classification used for a meta-entity whose instances can exist without the occurrences of other meta-entities (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)
key migration. (1) the modeling process of placing the primary key of a parent or generic entity in its child or category entity as a foreign key (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.96) Note: [key style]
key-style view. (1) view that represents the structure and semantics of data within an enterprise (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.97) Note: That is, data (information) models.
keyword. (1) one or more words used as a reference to a specific set of actions intended to be performed during the execution of one or more test cases (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.3) Note: Keywords are named using at least one verb. Composite keywords can be constructed based on other keywords.
keyword dictionary. (1) repository containing a set of keywords reflecting the language and level of abstraction used to write test cases (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.4) Syn: keyword library

keyword execution code. (1) implementation of a keyword that is intended to be executed by a test execution engine (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.7)

keyword test case. (1) sequence of keywords and the required values for their associated parameters (as applicable) that are composed to describe the actions of a test case (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.8)

keyword-driven testing. (1) testing using test cases composed from keywords (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.5)

keyword-driven testing framework. (1) test framework covering the functional capabilities of a keyword-driven editor, decomposer, data sequencer, manual test assistant, tool bridge, data and script repositories, a keyword library and the test execution environment (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.6)

kNN. (1) k-nearest neighbors (ISO/IEC/IEEE 24765:2021)

knowledge. (1) aspect of an instance's specification that is determined by the values of its attributes, participant properties, and constant, read-only operations (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.98) (2) a mixture of experience, values and beliefs, contextual information, intuition, and insight that people use to make sense of new experiences and information (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

knowledge base (K-base). (1) database that contains inference rules and information about human experience and expertise in a domain. (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: In self-improving systems, the knowledge base additionally contains information resulting from the solution of previously encountered problems

knowledge management. (1) multi-disciplinary process of obtaining, preserving, sharing, using, and refreshing knowledge (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: In DevOps, knowledge management guides and facilitates the automation of system operations, system problem identification and remediation, and reporting system health.

known error. (1) result of a problem with an identified root cause or an identified workaround that reduces or eliminates its impact (ISO/IEC/IEEE 24765c:2014)

KOPS. (1) kilo-operations per second; that is, thousands of operations per second (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: a measure of computer processing speed See also: MFLOPS, MIPS

KPI. (1) key performance indicator (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

label. (1) a name or identifier assigned to a computer program statement to enable other statements to refer to that statement (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) one or more characters, within or attached to a set of data, that identify or describe the data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) a word or phrase that is attached to, or part of, a model graphic (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.99) (4) information associated with the net graph or one of its objects (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.5) (5) item, attached to a product (if practicable) or its packaging, which displays information related to one or more characteristic(s) of the product (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.22)
lag. (1) the amount of time whereby a successor activity is required to be delayed with respect to a predecessor activity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
language. (1) systematic means of communicating ideas by the use of conventionalized signs, sounds, gestures, or marks and rules for the formation of admissible expressions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) means of communication, with syntax and semantics, consisting of a set of representations, conventions, and associated rules used to convey information (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
language processor. (1) computer program that translates, interprets, or performs other tasks required to process statements expressed in a given language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembler, compiler, interpreter, translator
language standard. (1) standard that describes the characteristics of a language used to describe a requirements specification, a design, or test data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
late binding. (1) the assignment of tasks to specific resources when the resources are available to start work, rather than when the project is planned (Software Extension to the PMBOK(R) Guide Fifth Edition)
late finish date (LF). (1) in the Critical Path Method, the latest possible point in time when the uncompleted portions of a schedule activity can finish based on the schedule network logic, the project completion date, and any schedule constraints (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
late start date (LS). (1) in the Critical Path Method, the latest possible point in time when the uncompleted portions of a schedule activity can start based on the schedule network logic, the project completion date, and any schedule constraints (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
latency. (1) time interval between the instant at which an instruction control unit issues a call for data and the instant at which the transfer of data is started (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
latent variable. (1) variable representing a unidimensional construct (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.8)
lateral compression. (1) in software design, a form of demodularization in which two or more modules that execute one after the other are combined into a single module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: downward compression, upward compression
layer. (1) partition resulting from the functional division of a software system, where layers are organized in a hierarchy; there is only one layer at each level in the hierarchy; there is a superior/subordinate hierarchical dependency between the functional services provided by software in any two layers in the software system that exchange data directly; and the software in any two layers in the software system that exchange data interpret only part of that data identically (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.15)
layout. (1) physical organization of source code including the use of white space, grouping, blank lines, alignment, indentation, and parentheses (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
LCC. (1) life-cycle cost (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
LCCE. (1) life-cycle cost estimate (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
LCMS. (1) Learning Content Management System (ISO/IEC/IEEE 26531:2015 Systems and software engineering --Content management for product lifecycle, user and service management documentation, 5)
LCSP. (1) life cycle sustainment plan (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)
lead. (1) the amount of time whereby a successor activity can be advanced with respect to a predecessor activity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: lag
leading decision. (1) loop control that is executed before the loop body (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: trailing decision, WHILE
leaf diagram. (1) diagram that has no descendent diagrams (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.68) Note: That is, a diagram that does not contain any function that has been decomposed.
leaf node. (1) function that is not decomposed (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.69) Note: A box that represents a leaf node does not have a box detail reference.
learnability. (1) degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.4.2) (2) degree to which an IT service can be learned by users to achieve a specified level of effectiveness, efficiency, freedom from risk and satisfaction within a
specified amount of time and context of use *(ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.2.2)* Note: Can be specified or measured either as the extent to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use, or by product properties corresponding to suitability for learning as defined in ISO 9241-110.

**left-shift.** *(1)* prioritizing the involvement of relevant stakeholders in applying quality activities, security, privacy, performance, verification, and validation earlier in the life cycle *(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)*

**legacy software.** *(1)* software originally created without information structures *(ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.17)*

**lessons learned.** *(1)* the knowledge gained during a project which shows how project events were addressed or should be addressed in the future with the purpose of improving future performance *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**lessons learned register.** *(1)* a project document used to record knowledge gained during a project so that it can be used in the current project and entered into the lessons learned repository *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**lessons learned repository.** *(1)* a project document used to record knowledge gained during a project so that it can be used in the current project and entered into the lessons learned repository *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**level.** *(1)* designation of the coverage and detail of a view *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.100)* Note: There are multiple levels of view; each is intended to be distinct, specified in terms of the modeling constructs to be used.

**level of abstraction.** *(1)* view of an object at a specific level of detail *(ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.12)*

**level of effort (LOE).** *(1)* an activity that does not produce definitive end products and is measured by the passage of time *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* Note: One of three EVM types of activities used to measure work performance


**level of service.** *(1)* parameters, or combination of parameters, which reflect social, political, environmental and economic outcomes that the organization delivers *(ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.31)*

**Levenshtein distance.** *(1)* measure of the difference between two character sequences based on the minimum number of single character edits (insertion, deletion, or substitution) needed to convert one word to the other.
(ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.24)

**lexicography.** (1) decision technique that prioritizes the decision attributes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dominance, satisficing

**LF.** (1) late finish date (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**LFT&E.** (1) live fire test and evaluation (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**liaison.** (1) relationship between a set of objects which results from the performance of some establishing behavior (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.2.4)

(2) state of having a contractual context in common (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.2.4)

**library.** (1) place containing collections of work products and useful information items for people to read, borrow or refer to, and for machines to access and retrieve data from (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.10) Note: In a repository, work products and other items are preserved for future retrieval when needed, whereas in a library, working data is temporarily stored and retrieved as necessary.

**license.** (1) legal agreement between two parties, the licensor and the licensee, as to the terms and conditions for the use or transfer of an intellectual property right from the licensor to the licensee (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**license compliance audit.** (1) audit that reconciles license-related information from multiple information sources, such as entitlement consumption against entitlement rights (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**license model.** (1) class of licenses with common characteristics (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**licensing standard.** (1) standard that describes the characteristics of an authorization given by an official or a legal authority to an individual or organization to do or own a specific thing (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
life cycle cost. (1) total investment in product development, manufacturing, test, distribution, operation, support, training, and disposal. (ISO/IEC/IEEE 24765:2016) (2) total cost of acquisition and ownership of a system over its entire life. (INCOSE Systems Engineering Handbook, 4th ed.) Note: It includes all costs associated with the system and its use in the concept, development, production, utilization, support, and retirement stages. Syn: life-cycle cost, LCC

life cycle model. (1) framework of processes and activities concerned with the life cycle that can be organized into stages, which also acts as a common reference for communication and understanding. (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.27) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.24) (2) framework containing the processes, activities, and tasks involved in the development, operation, and maintenance of a software product, spanning the life of the system from the definition of its requirements to the termination of its use. (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.1) Syn: life-cycle model

life cycle processes. (1) set of interrelated or interacting activities that result in the development or assessment of system, software, or hardware products. (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) Note: Each activity consists of tasks. The life cycle processes can overlap one another. For VV purposes, no process is concluded until its development products are verified and validated according to the defined tasks in the validation and verification plan. Syn: life-cycle processes

lightweight process. (1) process with a single thread of control; a task. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

likelihood. (1) probability of something happening. (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.3.6) (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 3.13) (2) chance of something happening. (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.2) Note: Refers to the chance of something happening, whether defined, measured, or determined objectively or subjectively, qualitatively or quantitatively, and described using general terms or mathematically (such as a probability or a frequency over a given time period).

limit. (1) restriction on rights or privileges granted by a software entitlement. (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.16)

limited entry table. (1) decision table where all the conditions and actions are completely described without reference to the rules. (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.14)

line of code. (1) programming-language statement; a non-comment, nonblank deliverable source statement. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

linear interpolation. (1) approximation of the value of a function at a given point, based on values on a straight line between two known points. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

link. (1) to create a load module from two or more independently translated object modules or load modules by resolving cross-references among them. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) part of a computer program, often a single instruction or address, which passes control and parameters between separate modules of the program. (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product
lifecycle, user and service management documentation, 4.25) (3) to provide a link as in (2) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) reference from some part of one document to some part of another document or another part of the same document (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.15) Syn: hyperlink See also: linkage editor

**linkage editor.** (1) computer program that creates a single load module from two or more independently translated object modules or load modules by resolving cross-references among the modules and, possibly, by relocating elements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: can be part of a loader Syn: linker See also: linking loader

**linking loader.** (1) computer program that reads one or more object modules into main memory in preparation for execution, creates a single load module by resolving cross-references among the separate modules, and, in some cases, adjusts the addresses to reflect the storage locations into which the code has been loaded (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: absolute loader, relocating loader, linkage editor

**list.** (1) set of data items, each of which has the same data definition (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to print or otherwise display a set of data items (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) collection class that contains no duplicates and whose members are ordered (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.101)

**list function.** (1) an on-line function that displays an overview of either all entity type occurrences or only those entity type occurrences that satisfy a certain selection criterion (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

**list processing language.** (1) programming language designed to facilitate the manipulation of data expressed in the form of lists (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: algebraic language, algorithmic language, logic programming language

**listing.** (1) ordered display or printout of data items, program statements, or other information (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**literal.** (1) in a source program, an explicit representation of the value of an item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) denotation of a specific instance of a value class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.102) (3) number or string that is used by a program directly rather than being embedded in a named constant or variable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**LITSR.** (1) level interim test status report (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**load.** (1) to read machine code into main memory in preparation for execution and, in some cases, to perform address adjustment and linking of modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to copy computer instructions or data from external storage to internal storage or from internal storage to registers (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: loader

**load map.** (1) computer-generated list that identifies the location or size of all or selected parts of memory-resident
load module. (1) computer program or subprogram in a form suitable for loading into main storage for execution by a computer; usually the output of a linkage editor (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: object module

load testing. (1) type of performance efficiency testing conducted to evaluate the behavior of a test item under anticipated conditions of varying load, usually between anticipated conditions of low, typical, and peak usage (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.20)

load-and-go. (1) operating technique in which there are no stops between the loading and execution phases of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

loaded origin. (1) address of the initial storage location of a computer program at the time the program is loaded into main memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembled origin, offset (1), starting address

loader. (1) computer program that reads machine code into main memory in preparation for execution and, in some cases, adjusts the addresses and links the modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) any program that reads programs or data into main memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include absolute loader, linking loader, relocating loader. See also: bootstrap, linkage editor

LOC. (1) lines of code (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 4)

local area network (LAN). (1) computer network located on a user's premises within a limited geographical area (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary can be subject to some form of regulation.

local compaction. (1) in microprogramming, compaction in which microoperations are not moved beyond the boundaries of the single-entry, single-exit sequential blocks in which they occur (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: global compaction

local customization. (1) FSM method that has been modified for local use, such that it might produce different functional sizes from those obtained prior to modification (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.9)

local data. (1) data that can be accessed by only one module or set of nested modules in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) data that can be accessed only within the routine in which it is declared (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: global data

local SAM owner. (1) individual at a level of the organization below that of the SAM owner who is identified as being responsible for SAM for a defined part of the organization (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.21)
local variable. (1) variable that can be accessed by only one module or set of nested modules in a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: global variable


localization. (1) creation of a national or specific regional version of a product (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.28) Note: Localization can be carried out separately from the translation process. Documentation can be localized even if the product has not been localized. See also: internationalization

location facility. (1) set of service primitives that allow a client-side binder object to ask a server-side if it will accept requests carrying invocations to a particular (computational) server object (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.9) Note: The server-side can confirm or reject the proposal or suggest an alternative server-side that is capable of handling requests.

location in space. (1) interval of arbitrary size in space at which an action can occur (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.10)

location in time. (1) interval of arbitrary size in time at which an action can occur (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.11)

location reference. (1) indicator following a heading or subheading in an index or table of contents, showing to which part of the document the heading or subheading refers (ISO/IEC/IEEE 24765a:2011)

location transparency. (1) distribution transparency which masks the use of information about location in space when identifying and binding to interfaces (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.4.1.3)

lock. (1) exclusive permission to edit a file (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

lockout. (1) computer resource allocation technique in which shared resources (especially data) are protected by permitting access by only one device or process at a time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: deadlock, semaphore

LOE. (1) level of effort (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

log. (1) a document used to record and describe or denote selected items identified during execution of a process or activity. Usually used with a modifier, such as issue, change, or assumption (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

log off. (1) to end a session (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: log out

log on. (1) to initiate a session (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: log in

logic programming language. (1) a programming language used to express programs in terms of control constructs and a restricted predicate calculus (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: algebraic language, algorithmic language, list processing language
logical cohesion. (1) type of cohesion in which the tasks performed by a software module perform logically similar functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: coincidental cohesion, communicational cohesion, functional cohesion, procedural cohesion, sequential cohesion, temporal cohesion

logical file. (1) a logical group of permanent data seen from the perspective of the user (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) logical group of data as seen by the user (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: A logical file is made up of one or more data entities. It is an internal logical file or an external interface file. See also: data function

logical layout. (1) the set of user required data element types and their logical structure as defined for an output product, apart from aspects of physical implementation (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: This does not pertain to the physical way in which data is presented on a screen, report, or other media.

logical record. (1) set of data which is processed in a single iteration of the main procedure (ISO/IEC/IEEE 24765a:2011) Note: It can be part or the whole of a single physical record or of a number of records.

logical relationship. (1) a dependency between two activities, or between an activity and a milestone (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: dependency See also: precedence relationship

logical source statement (LSS). (1) software instruction, independent of the physical format (lines of code) in which it appears (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: physical source statement

logical trace. (1) execution trace that records only branch or jump instructions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

logical transaction. (1) the basic functional component of Mk II FPA (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) Note: The smallest complete unit of information processing that is meaningful to the end user in the business. It is triggered by an event in the real world of interest to the user, or by a request for information. It comprises an input, process and output component. It must be self-contained and leave the application being counted in a consistent state.

logical type. (1) data type whose members can assume only logical values (usually TRUE and FALSE) and can be operated on only by logical operators, such as and, OR, and NOT (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: character type, enumeration type, integer type, real type

loop. (1) sequence of computer program statements that is executed repeatedly until a given condition is met or while a given condition is true (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to execute a sequence of computer program statements as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: iterative construct See also: loop body, loop control, UNTIL, WHILE

loop assertion. (1) logical expression specifying one or more conditions that must be met each time a particular point in a program loop is executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: loop
invariant See also: input assertion, output assertion, inductive assertion method

loop body. (1) part of a loop that accomplishes the loop's primary purpose (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: loop control

loop control. (1) part of a loop that determines whether to exit from the loop (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: loop body, leading decision, trailing decision

loop-control variable. (1) program variable used to determine whether to exit from a loop (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

loopback. (1) an internal arrow that is the output of a box whose box number is greater than the box number of the box that uses that arrow as input, control, or mechanism (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.70) Note: These uses are referred to as input loopback, control loopback, and mechanism loopback, respectively.

loopback testing. (1) testing in which signals or data from a test device are input to a system or component, and results are returned to the test device for measurement or comparison (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

low level. (1) specific; detailed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

low-level design. (1) process of design at the individual-routine or, sometimes, class level under the guidance of a more general design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: detailed design See also: detailed design

low-level keyword. (1) keyword that covers only one or very few simple actions and is not composed from other keywords (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.9)

low-profile quad flat package (LQFP). (1) semiconductor device based on a leadframe with gull wing-shaped leads on all four sides (ISO/IEC/IEEE 24765e:2015) Note: The LQFP package can be encapsulated in plastic.

lowclass. (1) if an instance is in a class S and not in any subclass of S, then S is the lowclass for the instance (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.103)


LRIP. (1) low-rate initial production (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

LS. (1) late start date (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

LTC. (1) level test case (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

LTD. (1) level test design (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

LTL. (1) level test log (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

LTP. (1) level test plan (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: LTPr

LTPr. (1) level test procedure (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: LTP

LTR. (1) level test report (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

MAC. (1) Media Access Control (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

machine code. (1) computer instructions and data definitions expressed in a form that can be recognized by the processing unit of a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembly code, compiler code, interpretive code

machine language. (1) language that can be recognized by the processing unit of a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Such a language usually consists of patterns of 1s and 0s, with no symbolic naming of operations or addresses. Syn: first-generation language, machine-oriented language See also: assembly language, fifth-generation language, fourth-generation language, high-order language, symbolic language

machine learning (ML). (1) process using computational techniques to enable systems to learn from data or experience (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.43)

machine-dependent. (1) pertaining to software that relies on features unique to a particular type of computer and therefore executes only on computers of that type (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: machine-independent

machine-independent. (1) pertaining to software that does not rely on features unique to a particular type of computer, and therefore executes on computers of more than one type (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: machine-dependent, portability

machine-readable. (1) pertaining to data in a form that can be automatically generated by and input to a computer (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Syn: machine readable

macro. (1) in software engineering, a predefined sequence of computer instructions that is inserted into a program, usually during assembly or compilation, at each place that its corresponding macroinstruction appears in the program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: macro definition See also: macroinstruction, macrogenerator, open subroutine

macro library. (1) collection of macros available for use by a macrogenerator (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: system library

macroassembler. (1) assembler that includes, or performs the functions of, a macrogenerator (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

macrogenerator. (1) routine, often part of an assembler or compiler, that replaces each macroinstruction in a source program with the predefined sequence of instructions that the macroinstruction represents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: macro-generating program

macroinstruction. (1) source code instruction that is replaced by a predefined sequence of source instructions, usually in the same language as the rest of the program and usually during assembly or compilation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: macro, macrogenerator
macroprocessor. (1) routine or set of routines provided in some assemblers and compilers to support the definition and use of macros (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

macroprogramming. (1) computer programming using macros and macroinstructions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

magic number. (1) literal value that is used by a program directly rather than being embedded in a named constant or variable (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: magic string See also: literal

magnetic core memory. (1) volatile memory that uses magnetic rings as the storage element (ISO/IEC/IEEE 24765c:2014)

main probe. (1) phase to perform repetitive cycle for gathering and analyzing data for finding strengths and challenges of an organization (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.2)

main procedure. (1) all those activities subsequent to the general initiation routine and prior to the general termination routine within the complete procedure (ISO/IEC/IEEE 24765a:2011)

main program. (1) software component that is called by the operating system of a computer and that usually calls other software components (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: routine, subprogram

mainframe. (1) computer intended to run in a computer center, with extensive capabilities and resources to which other computers can be connected so that they can share facilities (ISO/IEC 2382:2015 Information technology -- Vocabulary)

maintain. (1) add, change or delete data through an elementary process (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.40) (2) to add, change or delete data through a transactional function type (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) maintainability. (1) ease with which a software system or component can be modified to change or add capabilities, correct faults or defects, improve performance or other attributes, or adapt to a changed environment (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) ease with which a hardware system or component can be retained in, or restored to, a state in which it can perform its required functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) capability of the software product to be modified (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.4) (4) average effort required to locate and fix a software failure (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (5) speed and ease with which a program can be corrected or changed (IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, 2.3) (6) degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.7) Note: Maintainability includes installation of updates and upgrades. Modifications include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications. Modifications include those carried out by specialized support staff, and those carried out by business or operational staff, or end users. See also: extendability,
flexibility

**maintainability plan.** (1) document setting out the specific maintainability practices, resources and sequence of activities relevant to software *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: The developer prepares the Maintainability Plan.

**maintainability testing.** (1) test type conducted to evaluate the degree of effectiveness and efficiency with which a test item can be modified *(ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.21)*

**maintainer.** (1) individual or organization that performs maintenance activities *(ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.31)*

**maintenance.** (1) process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment *(ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.9)* (2) process of retaining a hardware system or component in, or restoring it to, a state in which it can perform its required functions *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* (3) actions intended to retain a product in, or restore it to, a useful and safe condition, in which it can perform the intended use *(IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.23)* See also: adaptive maintenance, corrective maintenance, perfective maintenance, software maintenance

**maintenance branch.** (1) branch where most development concerns bug fixes *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**maintenance enhancement.** (1) modification to an existing software product to satisfy a new requirement *(IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.5)* Note: There are two types of software enhancements, adaptive and perfective. A maintenance enhancement is not a software correction.

**maintenance manual (MM).** (1) software engineering project-deliverable document that enables a system’s maintenance personnel (rather than users) to maintain the system *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: operator manual, user manual

**maintenance personnel.** (1) software engineers who maintain software systems *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**maintenance plan.** (1) document setting out the specific maintenance practices, resources, and sequence of activities relevant to maintaining a software product *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**maintenance program.** (1) organizational structure, responsibilities, procedures, processes, and resources used for implementing the maintenance plan *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Syn: maintenance infrastructure

**maintenance project.** (1) software development project described as maintenance to correct errors in an original requirements specification, to adapt a system to a new environment, or to enhance a system *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*
**Systems and software engineering—Vocabulary**

**make or buy analysis.** (1) the process of gathering and organizing data about product requirements and analyzing them against available alternatives including the purchase or internal manufacture of the product (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: make-or-buy analysis

**make or buy decision.** (1) decisions made regarding the external purchase or internal manufacture of a product (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: make-or-buy decisions

**malfunction.** (1) behavior of a system or component that deviates from the specifications (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.11)

**manage.** (1) [requirements] provide storing and editing capabilities, tracking history of edition, versioning, author identification, change management, time stamping, user notification for content changes, security rights control (ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.2)

**manage communications.** (1) the process of creating, collecting, distributing, storing, retrieving and the ultimate disposition of project information in accordance to the Communications Management Plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**manage stakeholder engagement.** (1) the process of communicating and working with stakeholders to meet their needs and expectations, address issues, and foster appropriate stakeholder involvement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**manage team.** (1) the process of tracking team member performance, providing feedback, resolving issues, and managing team changes to optimize project performance (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**manageability.** (1) degree to which IT infrastructure management can attain and keep an application in its operational state (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.24) Note: Manageability involves the transparency and manageability of applications from an infrastructure point of view.

**managed network.** (1) network or set of networks established and controlled by one or more organizations to meet specific organizational or business needs (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.16)

**managed process.** (1) performed process that is planned and executed in accordance with policy; employs skilled people having adequate resources to produce controlled outputs; involves relevant stakeholders; is monitored, controlled, and reviewed; and is evaluated for adherence to its process description (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: performed process

**managed role.** (1) view of the management interface of an object which is being managed within an ODP system (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 14.4)

**managed website.** (1) site created and maintained based on organizational guidelines (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.17) Syn: managed web site
management. (1) system of controls and processes required to achieve the strategic objectives set by the organization's governing body (ISO/IEC/IEEE 21840:2019 Systems and software engineering--Guidelines for the utilization of ISO/IEC/IEEE 15288 in the context of system of systems (SoS), 3.1.5) (2) coordinated activities to direct and control an organization (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.18) Note: Management is subject to the policy guidance and monitoring set through corporate governance. Management can include establishing policies and objectives, and processes to achieve these objectives. Management is a set of activities performed by managers. Syn: MGT

management information. (1) knowledge concerning objects which are of relevance to management (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 14.3)

management notification. (1) event notification initiated by an object operating in a managed role (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 14.6)

management process. (1) activities that are undertaken in order to ensure that the software engineering processes are performed in a manner consistent with the organization's policies, goals, and standards (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

management reserve. (1) amount of the project budget withheld for management control purposes (ISO/IEC/IEEE 24765h:2019) Note: These are budgets reserved for unforeseen work that is within scope of the project. The management reserve is not included in the Performance Measurement Baseline.

management review. (1) systematic evaluation of a 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 used to achieve fitness for purpose (ISO/IEC/IEEE 24765i:2020)

management skills. (1) ability to plan, organize, direct, and control individuals or groups of people to achieve specific goals (ISO/IEC/IEEE 24765h:2019)

management system. (1) set of interrelated or interacting elements to establish policy and objectives and to achieve those objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.32) Note: A management system can address a single discipline or several disciplines. The system elements include the organization's structure, roles and responsibilities, planning, and operation. The scope of a management system may include the whole of the organization, specific and identified functions of the organization, specific and identified sections of the organization, or one or more functions across a group of organizations.

managerial independence. (1) of software quality assurance (SQA), situation in which the responsibility of the SQA effort is vested in an organization separate from the development and project management organizations (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2)

managing role. (1) view of an object which is performing managing actions (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 14.5)

mandatory. (1) syntax keyword used to specify a total mapping (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.104) Note: Mandatory requirements are expressed using "shall". Syn: MAN See also: optional, total
mandatory dependency. (1) a relationship that is contractually required or inherent in the nature of the work (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: hard logic

mandatory nonidentifying relationship. (1) nonidentifying relationship in which an instance of the child entity must be related to an instance of the parent entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.105) See also: optional nonidentifying relationship, nonidentifying relationship [key style]

manual testing. (1) humans performing tests by entering information into a test item and verifying the results (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.10) Note: Manual testing does not use tools, robots, or other test execution engines.

manufacture. (1) in software engineering, the process of copying software to disks, chips, or other devices for distribution to customers or users (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

manufacturing phase. (1) period of time in the software life cycle during which the basic version of a software product is adapted to a specified set of operational environments and is distributed to a customer base (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

many-sorted algebra. (1) mathematical structure comprising a set of sets and a set of functions taking these sets as domains and co-domains (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.14)

many-to-many relationship. (1) relationship between two state classes (not necessarily distinct) in which each instance of one class can be associated with any number of instances of a second class (possibly none), and each instance of the second class can be related to any number of instances of the first class (possibly none) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.106)

map program. (1) software tool, often part of a compiler or assembler, that generates a load map (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

mapping. (1) assigned correspondence between two things that is represented as a set of ordered pairs (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.107) (2) establishing a sequence of activities according to a selected software life cycle model (SLCM) (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: instance, invocation, iteration, software life cycle model (SLCM)

mapping completeness. (1) a designation of whether a mapping is complete (totally mapped) or incomplete (partial) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.108) See also: partial, total

mapping document. (1) document that relates a standard and an existing industry practice (ISO/IEC 19770-8:2020, Information technology IT asset management Part 8: Guidelines for mapping of industry practices to/from the ISO/IEC 19770 family of standards, 3.1)

marking. (1) symbols, pictograms, warnings, logos or inscriptions on the product, label, or packaging to identify its type, which can also include short textual messages (IEC/IEEE 82079-1:2019 Preparation of information for use
marking (of a net). (1) the set of the place markings for all places of the net (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.15)


markup language. (1) method of defining and describing the structure of different types of electronic documents (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.20)

MARR. (1) minimum attractive rate of return (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

mask. (1) a pattern of bits or characters designed to be logically combined with an unknown data item to retain or suppress portions of the data item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: interrupt mask

mask ROM. (1) read-only memory unit whose circuits are programmed during the manufacturing process (ISO/IEC/IEEE 24765c:2014)

master data. (Deprecated) (1) data held by an organization that describes the entities that are both independent and fundamental for an enterprise that it needs to reference in order to perform its transaction (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.25) Note: This term is deprecated. Syn: primary data

master library. (This term is deprecated.) (1) a software library containing master copies of software and documentation from which working copies can be made for distribution and use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: main library See also: production library, software development library, software repository, system library

master schedule. (This term is deprecated.) (1) a summary-level project schedule that identifies the major deliverables and work breakdown structure components and key schedule milestones (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: top-level schedule, main schedule, summary schedule See also: milestone schedule

material. (1) aggregate of things used by an organization in an undertaking (ISO/IEC/IEEE 24765h:2019) Syn: materiel

matrix diagrams. (1) a quality management and control tool used to perform data analysis within the organizational structure created in the matrix. The matrix diagram seeks to show the strength of relationships between factors, causes and objectives that exist between the rows and columns that form the matrix. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

matrix organization. (1) any organizational structure in which the project manager shares responsibility with the functional managers for assigning priorities and for directing the work of persons assigned to the project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

maturity. (1) degree to which a system, product or component meets needs for reliability under normal operation (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation)
The concept of maturity can be applied to quality characteristics to indicate the degree to which they meet required needs under normal operation.

**maturity level.** (1) point on an ordinal scale of organizational process maturity that characterizes the maturity of the organizational unit assessed in the scope of the maturity model used (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.1)

**maturity model.** (1) model derived from one or more specified process assessment model(s) that identifies the process sets associated with the levels in a specified scale of organizational process maturity (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.7)

**maximax rule.** (1) in decision making under uncertainty, assuming that the best state of nature will happen, selection of the alternative that has the best payoff from all of the best payoffs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: Hurwicz criterion, maximin rule, minimax regret rule

**maximin rule.** (1) in decision making under uncertainty, assuming that the worst state of nature will happen, selection of the alternative that has the best payoff from all of the worst payoffs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: the most pessimistic of the uncertainty techniques See also: Hurwicz criterion, maximax rule, minimax regret rule

**MBlA.** (1) manage benchmarking business level activity (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)

**MCDM.** (1) multiple-criteria decision making (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.9) (2) multiple condition decision method (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 4)

**MCU.** (1) microcontroller unit (ISO/IEC/IEEE 24765c:2014)

**MD5.** (1) message digest 5 (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.22)


**MDD.** (1) model driven development (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.2)

**mean execution time.** (1) the mean value of all execution times of tasks of the j-th task type which were submitted within the rating interval (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.8)

**mean execution time rating value.** (1) the quotient (corresponding to the j-th task type) of the mean execution time reference value and the measured mean execution time (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.9)

**mean execution time reference value.** (1) the mean execution time maximally accepted by the emulated user (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.1)

**mean time between failures (MTBF).** (1) the expected or observed time between consecutive failures in a
system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: up time

**mean time to repair (MTTR).** (1) expected or observed duration required to return a malfunctioning system or component to normal operations (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) the mean time the maintenance team requires to implement a change and restore the system to working order (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: down time

**meaning (of a responsibility).** (1) statement of what the responsibility means (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.109) Note: The statement of responsibility states what the requester needs to know to make intelligent use of the property or constraint. That statement is complete enough to let a requester decide whether to make the request, but it stops short of explaining how a behavior or value is accomplished or derived. Meaning is initially captured using freeform natural language text in a glossary definition. It can be more formally refined into a statement of pre-conditions and post-conditions using the specification language.

**meaningful.** (1) user-recognizable and satisfies a functional user requirement (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.41)


**measurand.** (1) particular quantity subject to measurement (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.5) Note: The specification of a measurand involves statements about quantities such as time, temperature and pressure.


**measure of effectiveness (MOE).** (1) operational measure of success that is closely related to the achievement of the operational objective being evaluated in the intended operational environment under a specified set of conditions (ISO/IEC/IEEE 24748-4:2016 Systems and software engineering-Life cycle management-Part 4: Systems engineering planning, 4.7) Note: Defines the information needs of the decision makers with respect to system effectiveness to meet operational expectations.

**measure of performance (MOP).** (1) engineering parameter that provides critical performance requirements to satisfy a measure of effectiveness (MOE) (ISO/IEC/IEEE 24748-4:2016 Systems and software engineering-Life cycle
management-Part 4: Systems engineering planning, 4.8) (2) measure that defines the key performance characteristics the system should have when fielded and operated in its intended operating environment (INCOSE Systems Engineering Handbook, 4th ed.) Note: An MOP typically characterizes physical or functional attributes relating to the system operation.

measurement. (1) set of operations having the object of determining a value of a measure (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.20) (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.17) (2) process to determine a value (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.34) (3) use of a metric to assign a value (e.g. a number or category) from a scale to an attribute of an entity (ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools) (4) assignment of values and labels to software engineering work products, processes, and resources plus the models that are derived from them, whether these models are developed using statistical or other techniques (ISO/IEC TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK), 7)

measurement analyst. (1) individual or organization that is responsible for the planning, performance, evaluation, and improvement of measurement (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.18)

measurement experience base. (1) data store that contains the evaluation of the information products and the measurement process as well as any lessons learned during the measurement process (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.19)

measurement function. (1) algorithm or calculation performed to combine two or more base measures (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.20) (2) algorithm or calculation performed to combine two or more quality measure elements (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.7)

measurement librarian. (1) individual or organization that is responsible for managing the measurement data store(s) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

measurement method. (1) logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.21) (2) logical organization of operations, described generically, used in measurement (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.8) (3) logical sequence of operations, described generically, used in the performance of measurements (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.16) Note: The type of measurement method depends on the nature of the operations used to quantify an attribute. Two types are distinguished: subjective - quantification involving human judgment; objective - quantification based on numerical rules.

measurement model. (1) implicit or explicit relationship between a latent variable and its (multi-item) measures (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology)

measurement procedure. (1) set of operations, described specifically, used in the performance of a particular measurement according to a given method (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.10)
process, 3.22) (2) logical organization of operations, applied specifically, used in the performance of particular measurements according to a given measurement method (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.9) Note: A measurement procedure is usually recorded in a document that is sometimes itself called a "measurement procedure" and is usually in sufficient detail to enable an operator to carry out a measurement without additional information.

measurement process. (1) process for establishing, planning, performing and evaluating software measurement within an overall project or organizational measurement structure (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.23) (2) process of establishing, planning, performing and evaluating software measurement within an overall project or organizational measurement structure (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.18) (3) process for establishing, planning, performing and evaluating systems and software measurement within an overall project or organizational measurement structure (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.21)

measurement process owner. (1) individual or organization responsible for the measurement process (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.24)

measurement sponsor. (1) individual or organization that authorizes and supports the establishment of the measurement process (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.25)

measurement standard. (1) standard that describes the characteristics of evaluating a process or product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

measurement user. (1) individual or organization that uses the information products (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.26)

measuring instrument. (1) device intended to be used to make measurements, alone or in conjunction with supplementary device(s) (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.6)

mechanism. (1) in an IDEF0 model, the means used by a function to transform input into output (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.71)

mechanism arrow. (1) arrow or arrow segment that expresses IDEF0 mechanism (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.72) Note: That is, an object type set whose instances are used by a function to transform input into output. The arrowhead of a mechanism arrow is attached to the bottom side of a box.

mechanism loopback. (1) loopback of output from one function to be mechanism for another function in the same diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.73)

member product. (1) product belonging to the product line (ISO/IEC 26550:2015 Software and systems engineering-Reference model for product line engineering and management, 3.15) See also: application

memory. (1) addressable storage space in a processing unit and all other internal storage that is used to execute instructions (ISO/IEC 2382:2015 Information technology -- Vocabulary)
memory capacity. (1) maximum number of items that can be held in a given computer memory; usually measured in words or bytes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: channel capacity, storage capacity

memory compaction. (1) storage allocation technique in which the contents of all allocated storage areas are moved to the beginning of the storage space and the remaining storage blocks are combined into a single block (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) storage allocation technique in which contiguous blocks of non-allocated storage are combined to form single blocks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: garbage collection

memory dump. (1) display of the contents of all or part of a computer's internal storage, usually in binary, octal, or hexadecimal form (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: change dump, dynamic dump, postmortem dump, selective dump, snapshot dump, static dump

memory map. (1) diagram that shows where programs and data are stored in a computer's memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

menu. (1) list displayed on a screen showing available functions from which a choice can be made (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) (2) a list of options displayed by a data processing system, from which the user can select an action to be initiated (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) (on-screen documentation) list of topics from which the user chooses (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.29)

menu by-pass. (1) in a menu-driven system, a feature that permits advanced users to perform functions in a command-driven mode without selecting options from the menus (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

menu structure. (1) the implementation of a dialog by means of a series of interrelated menus and screens (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

menu-driven. (1) pertaining to a system or mode of operation in which the user directs the system through menu selections (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: menu by-pass, command-driven

merge. (1) to combine different changes to the same file (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Many systems follow the optimistic strategy of combining all lines that do not conflict.

merge from current. (1) to merge changes from the current branch into the stable branch(es) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: To avoid disruptive changes in a stable branch, code changes are typically first introduced into the current (development) branch, tested, and then merged back.

message. (1) communication sent from one object to another (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.110) Note: Message encompasses requests to meet responsibilities as well as simple informative communications. See also: request

meta-. (1) prefix to a concept to imply definition information about the concept (ISO/IEC 15474-1:2002 Information
technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Specifically, used to designate the location of an object in the three model layers.

meta-attribute. (1) definition of a characteristic of a meta-entity or meta-relationship (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Instances of a meta-attribute occur in a model as data values.

meta-entity. (1) definition of a type of data object that occurs in CDIF models (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Specifically, a meta-entity represents a set of zero or more meta-attributes, stored together to represent a thing, event or concept that has instances in a model.

meta-meta-attribute. (1) definition of a characteristic of a meta-meta-entity or meta-meta-relationship (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Instances of a meta-meta-attribute occur in a metamodel as meta-data values.

meta-meta-entity. (1) a definition of the behavior and structure of meta-entities, meta-relationships, meta-attributes, or subject areas (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: i.e., a definition of the meta-object definitions used to describe information in models

meta-meta-relationship. (1) definition of a type of data object that occurs in CDIF metamodels (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Specifically, a meta-meta-relationship represents the definition of a relationship between instances of meta-meta-entities.

meta-object. (1) generic term for meta-entities, meta-relationships and meta-attributes (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

meta-object facility. (1) specification of the object management group for repositories of type information for arbitrary type systems (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 3.3.1) Syn: MOF

meta-relationship. (1) definition of a type of data object that occurs in CDIF models (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Specifically, a meta-relationship represents the definition of a relationship between meta-entities that has instances in a model. A meta-relationship can also define a set of zero or more meta-attributes, stored together to represent characteristics of a relationship between meta-entities.

metadata. (1) data that describe other data (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.29)

metalanguage. (1) language used to specify some or all aspects of a language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: stratified language, unstratified language

metamodel. (1) special kind of model that specifies the abstract syntax of a modeling language (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (2) metamodel Vm for a subset of IDEFobject is a view of the constructs in the subset that is expressed using those constructs such that there exists a valid instance of Vm that is a description of Vm itself (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.111) (3) model containing detailed definitions of the meta-entities, meta-relationships and meta-attributes whose instances appear in the model section of a CDIF transfer (ISO/IEC 15474-1:2002 Information technology

method engineer. (1) person who designs, builds, extends and maintains methodologies (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.10) Note: Method engineers create methodologies from metamodels via generation.

method standard. (1) standard that describes the characteristics of the orderly process or procedure used in the engineering of a product or performing a service (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

methodology. (1) a system of practices, techniques, procedures, and rules used by those who work in a discipline (A Guide to the Project Management Body of Knowledge (PMBOK® Guide) -- Sixth Edition) (2) specification of the process to follow together with the work products to be used and generated, plus the consideration of the people and tools involved, during an IBD development effort (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.2)

methodology element. (1) simple component of a methodology (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.6) Note: Usually, methodology elements include the specification of what tasks, activities, techniques, models, documents, languages and/or notations can or must be used when applying the methodology. Methodology elements are related to each other, comprising a network of abstract concepts. Typical methodology elements are Capture Requirements, Write Code for Methods (kinds of tasks), Requirements Engineering, High-Level Modeling (kinds of activities), Pseudo-code, Dependency Graphs (notations), Class, Attribute (kinds of model
building blocks), Class Model, Class Diagram, Requirements Specification (kind of work products).

**metric.** (1) quantitative measure of the degree to which a system, component, or process possesses a given attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) defined measurement method and the measurement scale (ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools) See also: software quality metric

**MF.** (1) measurement framework (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.2)

**MFLOPS.** (1) millions of floating point operations per second (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) megaflops, a unit of measure of processing performance equal to one million floating-point operations per second (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: a measure of computer processing speed See also: KOPS, MIPS

**MGT.** (1) management (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.18)

**micro code assembler.** (1) computer program that translates microprograms from symbolic form to binary form (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**microarchitecture.** (1) microword definition, data flow, timing constraints, and precedence constraints that characterize a given microprogrammed computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**microcode.** (1) collection of microinstructions, comprising part of, or all of, a set of microprograms (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) Syn: micro code

**microcomputer.** (1) digital computer whose processing unit consists of one or more microprocessors, and includes storage and input-output facilities (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: microprogrammable computer

**microcontroller (unit) (MCU).** (1) unit with application control function on a single chip (ISO/IEC/IEEE 24765c:2014) Note: It contains a processor, RAM, ROM, clock, register, and I/O control unit. Syn: single chip microcomputer

**microinstruction.** (1) in microprogramming, an instruction that specifies one or more of the basic operations needed to carry out a machine language instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include diagonal microinstruction, horizontal microinstruction, vertical microinstruction. See also: microcode, microoperation, microprogram

**microoperation.** (1) in microprogramming, one of the basic operations needed to carry out a machine language instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: microinstruction

**microprocessor.** (1) processor whose elements have been miniaturized into one or a few integrated circuits (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**microprogram.** (1) sequence of instructions, called microinstructions, specifying the basic operations needed to carry out a machine language instruction (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and
microprogrammable computer. (1) microprogrammed computer in which microprograms can be created or altered by the user (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

microprogrammed computer. (1) computer in which machine language instructions are implemented by microprograms rather than by hard-wired logic (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: microarchitecture, microcomputer, microprogrammable computer

microprogramming. (1) process of designing and implementing the control logic of a computer by identifying the basic operations needed to carry out each machine language instruction and representing these operations as sequences of instructions in a special memory called control store (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This method is an alternative to hard-wiring the control signals necessary to carry out each machine language instruction. Techniques include bit steering, compaction, residual control, single-level encoding, two-level encoding. See also: micro code, microinstruction, microprogram

microword. (1) addressable element in the control store of a microprogrammed computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

middleware. (1) software layer between an operating system and the software applications (ISO/IEC/IEEE 24765d:2015)

migratability. (1) ability to change the configuration, substituting one reference point of an object for another while the object is being used (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 15.4.2) See also: portability


migration. (1) moving a cluster to a different capsule (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.27)

migration transparency. (1) distribution transparency which masks from an object, the ability of a system to change the location of that object (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.4.1.4) Note: Migration is often used to achieve load balancing and reduce latency.

milestone. (1) significant point or event in a project, program, or portfolio (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) scheduled event used to measure progress (ISO/IEC/IEEE 24765e:2015)

milestone list. (1) a list identifying all project milestones and normally indicates whether the milestone is mandatory or optional (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

milestone review. (1) formal review of a work product and supporting evidence used to determine its acceptability for use in the next stage of development or for delivery (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.10) Note: The requirement for this form of review is normally specified in the project plan.

milestone schedule. (1) type of schedule that presents milestones with planned dates (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

mind-mapping. (1) technique used to consolidate ideas created through individual brainstorming sessions into a single map to reflect commonality and differences in understanding and to generate new ideas (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: mind mapping

minicomputer. (1) digital computer that is functionally intermediate between a microcomputer and a mainframe (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: Servers and network devices have generally replaced minicomputers

minimalism. (1) principle for the selection of information for users that supports task performance, troubleshooting, and problem resolution (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.22) (2) principle that information for use includes critical information and the least amount of other information needed to be complete (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.25)

minimax regret rule. (1) in decision making under uncertainty, selection of the alternative that minimizes the regret that one would have, if one chose the wrong alternative under each state of nature; that is, the alternative that has the smallest maximum regret (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: Hurwicz criterion, maximax rule, maximin rule

minimum attractive rate of return (MARR). (1) lowest rate of return at which an organization will consider investing; the interest rate used in business decision analysis (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: reflects a rate of return that the organization is confident it can achieve through typical activities See also: opportunity cost

minimum delay programming. (1) programming technique in which storage locations for computer instructions and data are chosen so that access time is minimized (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

minimum tasks. (1) those verification and validation tasks required for the integrity level assigned to the system, software, or hardware to be verified and validated (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (2) those tasks required for the integrity level assigned to the software to be tested (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

minimum viable product. (1) version of a work product with just enough features and requirements to satisfy early customers and provide feedback for future development (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)

MIPS. (1) million instructions per second (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: a measure of computer processing speed See also: KOPS, MFLOPS

mirror site. (1) duplicate copy of a master site maintained on a different host typically to provide redundancy, higher performance, or local access (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.18)

MIS. (1) management information system (ISO/IEC/IEEE 24765b:2013)

mistake. (1) human action that produces an incorrect result (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)
Note: The fault tolerance discipline distinguishes between a human action (a mistake), its manifestation (a hardware or software fault), the result of the fault (a failure), and the amount by which the result is incorrect (the error).

**mixed entry table.** (1) decision table whose stub consists of rows in which limited and extended entries are written (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.16)

**mixed mode.** (1) pertaining to an expression that contains two or more different data types (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: mixed type


**MMC(S).** (1) Multimedia Conferencing (System) (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

**MML.** (1) man-machine interface (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: user interface

**mobile device.** (1) portable computing device, typically having a wireless internet connection and a display screen with touch, pen, or keyboard input, and possibly auditory input and output features (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.26) Note: Design of mobile devices has to fulfill special usability needs due to their size and available features for input and output.

**mobility schema.** (1) specification putting constraints on the mobility of an object (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 16.4.1.1)

**mock object.** (1) temporary dummy objects created to aid testing until the real objects become available (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**mock-up.** (1) throw-away product (ISO/IEC/IEEE 24765i:2020) Note: It can be retained for verification or training, and as a record.

**mode.** (1) set of related features or functional capabilities of a product (ISO/IEC/IEEE 24765e:2015)

algorithm or calculation combining one or more base or derived measures with associated decision criteria (ISO/IEC/IEEE 15939:2017 Systems and software engineering -- Measurement process, 3.27) (9) abstract representation of an entity or collection of entities that provides the ability to portray, understand or predict the properties or characteristics of the entity or collection under conditions or situations of interest (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.13) (10) output of a machine learning algorithm trained with a training dataset that generates predictions using patterns in the input data (ISO/IEC TR 29119-11:2020, Software and systems engineering -- Software testing -- Part 11: Guidelines on the testing of AI-based systems, 3.1.46) Note: representation of the concepts or properties of an entity and governing principles is captured in architecture models


**model hierarchy. (1)** diagrams that correspond to the nodes of the hierarchical graph structure of an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.74)

**model kind. (1)** conventions for a type of modeling (ISO/IEC/IEEE 42010:2011 Systems and software engineering--Architecture description, 3.9)

**model layers. (1)** different layers of definition (or abstraction) used in defining the CDIF family of standards (ISO/IEC 15474:1-2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The four model layers in CDIF are user data, model, metamodel, meta-metamodel. Any given model layer provides an accurate and complete definition of all the instances that occur one layer below the given layer. For example, the meta-metamodel provides a set of definitions that are used to construct and understand the metamodel; the metamodel provides a set of definitions that are used to construct and understand a model.

**model name. (1)** unique, descriptive name that distinguishes one IDEF0 model from other IDEF0 models with which it is associated (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.75) Note: An IDEF0 model's model name and model name abbreviation are placed in the A-0 context diagram along with the model's purpose statement and viewpoint statement.

**model name abbreviation. (1)** unique short form of a model name that is used to construct diagram references (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.76)

**model note. (1)** textual and/or graphical component of a diagram that records a fact not otherwise depicted by a diagram's boxes and arrows (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.77)

**model note number. (1)** integer number, placed inside a small square, that unambiguously identifies a model note in a specific diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.78)

**model page. (1)** logical component of an IDEF0 model that can be presented on a single sheet of paper (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.79) Note:
Model pages include diagram, text, FEO, and glossary pages.

**modeling.** (1) the activity of representing some elements of a process, device, or concept (Software Extension to the PMBOK(R) Guide Fifth Edition)

**modeling tool.** (1) tool that provides support for modeling, i.e., representing, a software product or an information system (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

**modifiability.** (1) ease with which a system can be changed without introducing defects (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.7.4) (3) degree to which an IT service can be effectively and efficiently modified without introducing defects or degrading existing IT service quality (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.8.2) See also: analyzability, maintainability, modularity

**modifiable.** (1) structured and has a style such that changes can be made completely, consistently, and correctly while retaining the structure (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.15)

**modification request (MR).** (1) proposed changes to a product that is being maintained (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.6) (2) forms associated with the various trouble/problem-reporting documents and the configuration change control documents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The MR can be classified as a correction or enhancement and identified as corrective, preventive, adaptive, or perfective maintenance. Trouble/problem-reporting documents include incident and problem reports; configuration change control documents include software change requests (SCRs). See also: change request, request for change

**modified source statement.** (1) source statement that has been changed from the original source (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**modified-off-the-shelf (MOTS).** (1) software product that is already developed and available, usable either 'as is' or with modification, and provided by the supplier, acquirer, or a third party (IEEE 1062-2015 IEEE Recommended Practice for Software Acquisition, 3.1)

**MODL.** (1) Meta-Object Definition Language (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 4)

**modular.** (1) composed of discrete parts (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: modular decomposition, modular programming

**modular decomposition.** (1) process of breaking a system into components to facilitate design and development; an element of modular programming (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: modularization See also: cohesion, coupling, demodularization, factoring, functional decomposition, hierarchical decomposition, packaging

**modular programming.** (1) software development technique in which software is developed as a collection of
modules *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transaction analysis, transform analysis

**modularity.** *(1)* degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components *(ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.7.1)* *(2)* software attributes that provide a structure of highly independent components *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: cohesion, coupling, modifiability

**module.** *(1)* program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading *(ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)* *(2)* logically separable part of a program *(ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)* *(3)* set of source code files under version control that can be manipulated together as one *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(4)* collection of both data and the routines that act on it *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: The terms 'module', 'component,' and 'unit' are often used interchangeably or defined to be subelements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized.

**module data.** *(1)* data that can be accessed by any routine within the module in which it is declared but not by routines in other modules *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Syn: instance data, class data

**MOE.** *(1)* measure of effectiveness *(IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)*


**monadic selective construct.** *(1)* if-then-else construct in which processing is specified for only one outcome of the branch, the other outcome resulting in skipping this processing *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: dyadic selective construct

**monitor.** *(1)* software tool or hardware device that operates concurrently with a system or component and supervises, records, analyzes, or verifies the operation of the system or component *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* collect project performance data with respect to a plan, produce performance measures, and report and disseminate performance information. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* Syn: execution monitor See also: hardware monitor, software monitor

**monitor and control project work.** *(1)* the process of tracking, reviewing, and regulating the progress to meet the performance objectives defined in the project management plan *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**monitor communications.** *(1)* the process of ensuring that the information needs of the project and its
stakeholders are met. (A Guide to the Project Management Body of Knowledge (PMBOK® Guide) -- Sixth Edition)

monitor risks. (1) the process of monitoring the implementation of agreed-upon risk response plans, tracking identified risks, identifying and analyzing new risks, and evaluating risk process effectiveness throughout the project. (A Guide to the Project Management Body of Knowledge (PMBOK® Guide) -- Sixth Edition)


monitoring. (1) determining the status of a system, a process, or an activity. (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.35)

monitoring and controlling process group. (1) those processes required to track, review, and regulate the progress and performance of the project; identify any areas in which changes to the plan are required; and initiate the corresponding changes. (A Guide to the Project Management Body of Knowledge (PMBOK® Guide) -- Sixth Edition)


Monte Carlo simulation. (1) an analysis technique where a computer model is iterated many times, with the input values chosen at random for each iteration driven by the input data, including probability distributions and probabilistic branches. Outputs are generated to represent the range of possible outcomes for the project. (A Guide to the Project Management Body of Knowledge (PMBOK® Guide) -- Sixth Edition)

MOP. (1) measure of performance. (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)

MOPS. (1) million operations per second. (ISO/IEC/IEEE 24765c:2014)

MOTS. (1) modified-off-the-shelf. (IEEE 1062-2015 IEEE Recommended Practice for Software Acquisition, 3.1)

MOU. (1) memorandum of understanding. (ISO/IEC/IEEE 24765c:2014)

move. (1) to read data from a source, altering the contents of the source location, and to write the same data elsewhere in a physical form that can differ from that of the source. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: copy

MPa. (1) measure IT project activity. (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)


MPLa. (1) manage benchmarking program level activity. (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)

MRa. (1) maintain repository activity. (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)

MSS. (1) management system standard. (ISO/IEC 29110-4-3:2018 Systems and software engineering-Lifecycle profiles for very small entities (VSEs)-Part 4-3: Service delivery-Profile specification, 4)

MT. (1) machine translation. (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers
**MTBF. (1)** mean time between failures (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**MTP. (1)** master test plan (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**MTR. (1)** master test report (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**MTTD. (1)** mean time to detection (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

**MTTR. (1)** mean time to repair (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) mean time to recovery (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

**multi-attribute decision. (1)** decision that considers more than just one criterion (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: multiple-attribute decision


**multi-core. (1)** chip with two or more microprocessor units (ISO/IEC/IEEE 24765e:2015)

**multi-core processor. (1)** single integrated circuit chip with more than one processing unit (ISO/IEC/IEEE 24765c:2014)

**multi-criteria decision analysis. (1)** This technique utilizes a decision matrix to provide a systematic analytical approach for establishing criteria, such as risk levels, uncertainty, and valuation, to evaluate and rank many ideas. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**multi-level cache. (1)** layered cache with progressively larger size and slower access (ISO/IEC/IEEE 24765c:2014)

*Note:* denoted by L1 (level 1) to L4 (level 4) cache, where the first level is the fastest and smallest memory size for immediate access, and the highest level is slowest, but the largest memory size

**multi-valued. (1)** a mapping that is not a function (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.117) See also: function


**multiaddress instruction. (1)** computer instruction that contains more than one address field (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: multiple-address instruction See also: one-address instruction

**multidimensional construct. (1)** construct that consists of a number of unidimensional constructs (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.11)

**multiple inclusive selective construct. (1)** special instance of the case construct in which two or more different values of the control expression result in the same processing (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
multiple inheritance. (1) ability of a subclass to inherit responsibilities from more than one superclass (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.119) (2) situation when the subtype inherits all of the meta-attributes and meta-relationships of all of its supertypes (and their supertypes) (ISO/IEC 15474-2:2002 Information technology -- CDIF framework -- Part 2: Modelling and extensibility, 6.2.6)

multiple instance approach. (1) case where each method of delivery of the same functionality is counted separately (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) See also: single instance approach

multiple readers and writers. (1) algorithm that lets multiple readers access a shared data repository concurrently; however, writers must have mutually exclusive access to update the repository (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

multiple-criteria decision making (MCDM). (1) making preference decisions (e.g., evaluation, prioritization, and selection) of available alternatives characterized by multiple criteria (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.9) Note: An MCDM with one alternative is the same as the development of a composite measure. Syn: multi-attribute decision making

multiple-hit decision table. (1) decision table where at least one set of conditions will be satisfied by more than one rule (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.3)


multiplicity. (1) natural number (i.e., non-negative integer) which describes the number of repetitions of an item in a multiset (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.18)

multiprocessing. (1) mode of operation in which two or more processes are executed concurrently by separate processing units that have access (usually) to a common main storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: multiprogramming, multitasking, time sharing

multiprogramming. (1) mode of operation in which two or more computer programs are executed in an interleaved manner by a single processing unit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: multiprocessing, multitasking, time sharing


multiset cardinality. (1) sum of the multiplicities of each of the members of the multiset (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.19) Syn: cardinality of a multiset

multitasking. (1) mode of operation in which two or more tasks are executed in an interleaved manner (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: multiprocessing, multiprogramming, time sharing

mutable class. (1) class for which the set of instances is not fixed; its instances come and go over time (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.119)
mutation testing. (1) testing methodology in which two or more program mutations are executed using the same test cases to evaluate the ability of the test cases to detect differences in the mutations (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

mutual exclusion. (1) giving access to shared data only to one task at a time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: can be enforced by means of binary semaphores or by using monitors.

mutually exclusive clustering. (1) task structuring criterion in which a group of objects are combined into one task because only one object can be executed at any one time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

N 2 diagram. (1) system engineering or software engineering tool for tabulating, defining, analyzing, and describing functional interfaces and interactions among system components (ISO/IEC/IEEE 24765e:2015) Note: The N 2 diagram is a matrix structure that graphically displays the bidirectional interrelationships between functions and components in a given system or structure. Syn: N2 diagram

n-address instruction. (1) computer instruction that contains n address fields, where n is any non-negative integer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-address instruction, two-address instruction, n-plus-one address instruction

N-ary relationship. (1) relationship with arity (degree) n &gt; 2 (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: A relationship that has more than two participating entities. (Note that a single entity can participate several times in a single relationship.)

n-level address. (1) indirect address that specifies the first of a chain of n storage locations, the first n-1 of which contains the address of the next location in the chain and the last of which contains the desired operand (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: direct address, immediate data

n-plus-one address instruction. (1) computer instruction that contains n+1 address fields, the last containing the address of the instruction to be executed next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-plus-one address instruction, two-plus-one address instruction, n-address instruction


N/R. (1) not required (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)

N2 diagram. (1) graphical representation used to define the internal operational relationships or external interfaces of the system of interest (INCOSE Systems Engineering Handbook, 4th ed.) Syn: N-squared diagram

name. (1) word or phrase that designates some model construct (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.121) (2) term which, in a given naming context, refers to an entity (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.1) Note: Such as a class, responsibility, subject domain, etc.

name resolution. (1) process by which, given an initial name and an initial naming context, an association between a
name and the entity designated by the initial name can be found (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.8)

**name space.** (1) set of terms usable as names (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.3)

**named constant.** (1) identifier that refers to a numeric or string value that does not change during program execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**named constraint.** (1) constraint that is specific to a particular model, rather than being inherent in some modeling construct (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.122) Note: Such as a cardinality constraint. A named constraint is explicitly named, its meaning is stated in natural language, and its realization is written in the specification language.

**naming action.** (1) action that associates a term from a name space with a given entity (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.5)

**naming context.** (1) relation between a set of names and a set of entities (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.4)

**naming domain.** (1) subset of a naming context such that all naming actions are performed by the controlling object of the domain (the name authority object) (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.6)

**naming graph.** (1) directed graph where each vertex denotes a naming context, and where each edge denotes an association between a name appearing in the source naming context and the target naming context (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 12.7)

**nano code.** (1) collection of nanoinstructions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**nanoinstruction.** (1) in a two-level implementation of microprogramming, an instruction that specifies one or more of the basic operations needed to carry out a microinstruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**nanostore.** (1) in a two-level implementation of microprogramming, a secondary control store in which nanoinstructions reside (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**narrow artificial intelligence.** (1) artificial intelligence focused on a single well-defined task to address a specific problem (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.47) Syn: narrow AI, weak artificial intelligence, week AI

**natural language.** (1) language whose rules are based on usage rather than being pre-established prior to the language's use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) language whose rules are based on current usage without being specifically prescribed (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: formal language

**navigation.** (1) means by which a user moves from one part of a software application to another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of accessing on-screen documentation and moving between different items of information (ISO/IEC/IEEE 26513:2017 Systems and software engineering--
Requirements for testers and reviewers of information for users, 3.27) (3) process of accessing on-screen information by moving between different locations in a website or electronic document (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.19)

navigational aids. (1) features of software that help the user to navigate around a computer application (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

NDI. (1) non-developmental item (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.2)

near-critical activity. (1) a schedule activity that has low total float. The concept of near-critical is equally applicable to a schedule activity or schedule network path. The limit below which total float is considered near critical is subject to expert judgment and varies from project to project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

negotiation. (1) resolution of disputes through consultations between involved parties (ISO/IEC/IEEE 24765h:2019)

nest. (1) to incorporate a computer program construct into another construct of the same kind (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary)

nesting. (1) embedding one construct inside another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

network. (1) arrangement of nodes and interconnecting branches (ISO/IEC 2382:2015 Information technology -- Vocabulary)

network chart. (1) directed graph used for describing and scheduling events, activities, and their relationships in project control (ISO/IEC 2382:2015 Information technology -- Vocabulary)

network logic. (1) all activity dependencies in a project schedule network diagram (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

network open end. (1) schedule activity without any predecessor activities or successor activities, causing a break in a schedule network path (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Network open ends are usually caused by missing logical relationships

network path. (1) sequence of activities connected by logical relationships in a project schedule network diagram (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

network planning. (1) technique that uses network charts for planning, scheduling, and controlling a project (ISO/IEC 2382:2015 Information technology -- Vocabulary)

networking. (1) developing relationships with persons who can assist in the achievement of objectives and responsibilities (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) establishing connections and relationships with other people from same or other organizations (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

neural network. (1) network of primitive processing elements connected by weighted links with adjustable weights, in which each element produces a value by applying a nonlinear function to its input values, and transmits it to other elements or presents it as an output value (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.48) Note: Whereas some neural networks are...
intended to simulate the functioning of neurons in the nervous system, most neural networks are used in artificial intelligence as realizations of the connectionist model. Syn: artificial neural network

**neuron coverage.** (1) proportion of activated neurons divided by the total number of neurons in the neural network (normally expressed as a percentage) for a set of tests (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.49) Note: A neuron is considered to be activated if its activation value exceeds zero.

**new source statements.** (1) sum of the added and modified source statements (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**NFR.** (1) non-functional requirement (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)

**NFSSM.** (1) non-functional software size measurement (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)

**no-op.** (1) no-operation (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**no-operation.** (1) computer operation whose execution has no effect except to advance the instruction counter to the next instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: used to reserve space in a program or, if executed repeatedly, to wait for a given event; often abbreviated no-op Syn: do-nothing operation

**node.** (1) in a diagram, a point, circle, or other geometric figure used to represent a state, event, or other item of interest (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) configuration of engineering objects forming a single unit for the purpose of location in space, and which embodies a set of processing, storage and communication functions (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.7) (3) modeled function located within the hierarchical graph structure of an IDEF0 model by its designated node number (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.80) (4) a point at which dependency lines connect on a schedule network diagram (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (5) vertex of a net graph, i.e. either a place or a transition (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.20) Note: A node can have internal structure which is not of concern in an engineering specification. See also: graph (2)

**node index.** (1) text listing, often indented, of the nodes in an IDEF0 model, shown in outline order (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.81) Note: Same meaning and node content as a node tree.

**node letter.** (1) letter that is the first character of a node number (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.82)

**node number.** (1) expression that unambiguously identifies a function's position in a model hierarchy (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.83) Note: A node number is constructed by concatenating a node letter, the diagram number of the diagram that contains the box that represents the function, and the box number of that box.

**node tree.** (1) graphical listing of the nodes of an IDEF0 model, showing parent-child relationships as a graphical tree
nomenclature standard. (1) standard that describes the characteristics of a system or set of names, or designations, or symbols (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

nominal group technique. (1) a technique that enhances brainstorming with a voting process used to rank the most useful ideas for further brainstorming or for prioritization (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

nominal scale. (1) scale in which the measurement values are categorical (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: ordinal scale, interval scale, ratio scale

non-compensatory model. (1) multiple-criteria decision-making model that does not allow criteria to compensate for each other in proportion to their weights (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.12) Note: Strongly positive or negative terms influence the overall composite value disproportionately, although the weight stays the same. There are various non-compensatory models depending on the evaluation policy, the purpose of the composite measure, or the measurement scale.

non-deliverable item. (1) hardware or software product that is not required to be delivered under the contract, but may be employed in the development of a product (ISO/IEC/IEEE 24765g:2018) Syn: nondeliverable item

non-deterministic system. (1) system which, given a particular set of inputs and starting state, will not always produce the same set of outputs and final state (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.50) Syn: nondeterministic system

non-functional requirement. (1) requirement for a software-intensive system or for a software product, including how it should be developed and maintained, and how it should perform in operation, except any functional user requirement for the software (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: Non-functional requirements (NFR) concern the software system or software product quality, the environment in which the software system or software product must be implemented and which it must serve, and the processes and technology to be used to develop and maintain the software system or software product and the technology to be used for their execution. See also: functional user requirement

non-functional size. (1) size of the software derived by quantifying the non-functional requirements (NFR), defined by a set of rules (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Syn: nonfunctional size

non-primary entity. (1) a data entity-type arrived at by Third Normal Form analysis which is not one of the main entity-types for which the application in question has been built (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) Note: Non-primary entities have only very few attributes, e.g. code, description See also: system entity

non-repudiation. (1) degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6.3)
non-terminal symbol. (1) part of the hierarchical definition of a syntax that is further decomposed in the hierarchy (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)


non-volatile memory. (1) unit that stores data whether power is on or off (ISO/IEC/IEEE 24765c:2014)

noncompensatory decision technique. (1) a multi-attribute decision technique that weighs all attributes equally, without allowing lower performance in one attribute to be traded off against better performance in another attribute (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) See also: compensatory decision technique


nondelivered source statement. (1) source statement that is developed in support of the final product, but not delivered to the customer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: non-delivered source statement

nondestructive read. (1) read operation that does not erase the data in the accessed location (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: non-destructive read See also: destructive read

nondeveloped source statement. (1) existing source statement that is reused or deleted (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: non-developed source statement

nondevelopmental. (1) developed prior to its current use in an acquisition or development process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Such an item can require minor modifications to meet the requirements of its current intended use.

nondimensional scaling. (1) decision technique in which attribute values are converted into a common scale where they can be added together to make a composite score for each alternative (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: additive weighting, analytic hierarchy process, compensatory decision technique

nonfunctional requirement. (1) software requirement that describes not what the software will do but how the software will do it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: design constraint, non-functional requirement, performance requirement, performance attribute See also: functional requirement, quality requirement

nonidentifying relationship. (1) a specific (not many-to-many) relationship in which some or all of the attributes contained in the primary key of the parent entity do not participate in the primary key of the child entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDefobject), 3.1.123) Syn: non-identifying relationship See also: identifying relationship, mandatory nonidentifying relationship, optional nonidentifying relationship [key style]

nonkey attribute. (1) attribute that is not the primary or a part of a composite primary key of an entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.125) Note: [key style] Syn: non-key attribute
nonprocedural language. (1) language in which the user states what is to be achieved without having to state specific instructions that the computer must execute in a given sequence (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: procedural language, declarative language, interactive language, rule-based language
nonprocedural programming language. (1) computer programming language used to express the parameters of a problem rather than the steps in a solution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: non-procedural programming language See also: procedural programming language
nontechnical requirement. (1) requirement affecting product and service acquisition or development that is not a property of the product or service (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
NOR. (1) in configuration management, a notice of revision (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
normalization. (1) process by which a data structure can be transformed by a database designer into a set of relations that have no repeating groups (ISO/IEC/IEEE 24765a:2011)
not printable. (1) not a <GeneralPrintableChar>, ", #, ], <EscapeCharacter> or <WhiteSpace> (ISO/IEC 15475-3:2002 Information technology -- CDIF transfer format -- Part 3: Encoding ENCODING.1, 7.2.11)
notation. (1) means of concrete representation for a particular type of a model, expressed as a grammar and suitable glyphs for its terminal symbols (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 7.5)
notation standard. (1) standard that describes the characteristics of formal interfaces within a profession (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
note. (1) helpful hint or other information that assists the user by emphasizing or supplementing important points of the main text (ISO/IEC/IEEE 24765g:2018) (2) body of free text that describes some general comment or specific constraint about a portion of a model (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.126) See also: caution, danger, warning
notebook computer. (1) battery-powered portable computer small and light enough to be operated anywhere (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: laptop computer
notice of revision (NOR). (1) form used in configuration management to propose revisions to a drawing or list, and, after approval, to notify users that the drawing or list has been, or will be, revised accordingly (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control, engineering change, specification change notice
nth of a kind. (1) re-manufacturing or re-installation of a previously verified and validated hardware or software design (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) Note: The nth of a kind component or system is equivalent to the first application in all relevant aspects, including functional and performance requirements, design documentation,
environment, and regulatory constraints.

**nucleus.** (1) engineering object which coordinates processing, storage and communications functions for use by other engineering objects within the node to which it belongs. (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.6) See also: kernel

**number of data element types (DETs).** (1) sum of all data element types (DETs) which are part of the input/output/query of the elementary process (EP), plus the data elements which are read or updated internally to the boundary. (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Syn: number of DETs

**numeric.** (1) pertaining to data that consists of numerals as well as functional units that use the data. (ISO/IEC 2382:2015 Information technology -- Vocabulary)


**OASIS.** (1) Organization for the Advancement of Structured Information Standards. (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)


**object code.** (1) computer instructions and data definitions in a form output by an assembler or compiler. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: An object program is made up of object code. See also: source code

**object identifier.** (1) some concrete representation for the identity of an object (instance). (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.128) Note: The object identifier (OID) is used to show examples of instances with identity, to formalize the notion of identity, and to support the notion in programming languages or database systems. Syn: OID

**object implementation.** (1) definition that provides the information needed to create an object and to allow the object to participate in providing an appropriate set of services. (ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.4.2)

Object Management Group (OMG). (1) international standards organization that owns and maintains CORBA(R)

---

**Obtain ISO Standards:** http://www.iso.org/iso/home/store/catalogue_ics.htm

**Obtain IEEE Standards:** http://www.techstreet.com/ieee

**Obtain PMI Standards:** http://marketplace.pmi.org/Pages/default.aspx?Category=PMBOKguide

This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
and UML(R) standards (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**object module. (1)** computer program or subprogram that is the output of an assembler or compiler (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: load module, object program

**object of interest (-type). (1)** any thing that is identified from the point of view of the functional user requirements about which the software is required to process or store data (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.19) Note: An object of interest can be any physical thing, as well as any conceptual object or part of a conceptual object in the world of the functional user. Syn: object of interest type

**object program. (1)** computer program that is the output of an assembler or compiler (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: target program See also: source program object module


**object set. (1)** subset of instantiations from the set of all possible instantiations of all object types within an object type set (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.86) Note: An object set is a subset of the union of the members of an object type set; the set of object sets includes the empty set and the set of the union of the members of the object type set itself. An object set is modeled by an arrow segment.

**object type. (1)** set of all possible instantiations of a singular concept, either physical or data, within an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.87)

(2) type whose members are object references (ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3.4) Note: An IDEF0 object type is generally analogous to an IDEF1X entity or an IDEF1 entity class.

**object type set. (1)** named set of one or more object types (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.88) Note: An object type set can include object types that are themselves grouped as object type sets. An object type set is designated by an arrow label.

**object-oriented design. (1)** software development technique in which a system or component is expressed in terms of objects and connections between those objects (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data structure-centered design, input-process-output, modular decomposition, rapid prototyping, stepwise refinement, structured design, transaction analysis, transform analysis

**object-oriented language. (1)** programming language that allows the user to express a program in terms of objects and messages between those objects (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**objective. (1)** something toward which work is to be directed, a strategic position to be attained, or a purpose to be achieved, a result to be obtained, a product to be produced, or a service to be performed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) practical advantage or intended effect, expressed as preferences about future states (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.2.1) (3) result to be achieved (ISO/IEC 19770-1:2017 Information
technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.37) (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.3) Note: An objective can be strategic, tactical, or operational. An objective can relate to different disciplines (such as financial, health and safety, and environmental goals) and can apply at different levels (such as strategic, organization-wide, project, product, and process). An objective can be expressed in other ways, e.g. as an intended outcome, a purpose, an operational criterion, an objective related to risk management, or by the use of other words with similar meaning (e.g. aim, goal, or target). Objectives related to risk management are set by the organization, consistent with the risk policy, to achieve specific results. Some objectives are ongoing; some are achieved once met. Syn: purpose

**objective evidence.** (1) data supporting the existence or verity of something (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.13) Note: Objective evidence can be obtained through observation, measurement, test, or other means. [ISO 9000:2015]

**objective function.** (1) formula that relates a decision variable to either the cost or the revenue of an alternative (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: cost function, income function


**obligation.** (1) prescription that a particular behavior is required (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.4)


**observation.** (1) instance of applying a measurement procedure to produce a value for a base measure (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.28)

**observation period.** (1) time interval, where the measurement procedure is observed for collecting (logging) measurement results for rating or validation, consisting of the rating interval and the supplementary run (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.11)

**occupational title standard.** (1) standard that describes the characteristics of the general areas of work or profession (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**OCD.** (1) operational concept document (IEC 1012-2016 IEC Standard for System, Software, and Hardware Verification and Validation, 3.2) See also: concept of operations (ConOps) document

**OCL.** (1) Object Constraint Language (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 4) (ISO/IEC 19793:2015 Information technology -- Open Distributed Processing -- Use of UML for ODP system specifications, 4)

**octet.** (1) byte that consists of eight bits (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: 8-bit byte

**ODP.** (1) Open Distributed Processing (ISO/IEC 19793:2015 Information technology -- Open Distributed Processing -- Use of UML for ODP system specifications, 4)

**ODP function.** (1) function required to support Open Distributed Processing (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.3.1)
ODP IDL. (1) Open Distributed Processing Interface Definition Language (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 4)

ODP standard. (1) standard that complies with the ODP Reference Model, directly or indirectly (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.3)

ODP system. (1) system which conforms to the requirements of ODP standards (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.5)

ODP-RM. (1) Open Distributed Processing: Reference Model (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 4) Syn: RM-ODP

OEM. (1) original equipment manufacturer (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.2)

off-the-shelf. (1) already developed and available (ISO/IEC/IEEE 24765g:2018)

office automation (OA). (1) integration of office activities by means of an information processing system (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: This term includes in particular the processing and communication of text, images, and voice.

offline. (1) pertaining to a device or process that is not under the direct control of the central processing unit of a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to the operation of a functional unit that takes place either independently of, or in parallel with, the main operation of a computer (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: online

offset. (1) difference between the loaded origin and the assembled origin of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) number that must be added to a relative address to determine the address of the storage location to be accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: relocation factor


OJT. (1) on-the-job-training (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)

OLA. (1) operations level agreement (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

OM. (1) organizational management (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.2)


on-chip oscillator. (1) electronic circuit on a microcomputer that produces a periodic electronic signal, often used for

**on-demand scheduling.** (1) a scheduling approach in which work is pulled from a backlog according to the perceived value to customers and is assigned as resources become available (Software Extension to the PMBOK(R) Guide Fifth Edition) See also: late binding

**on-screen documentation.** (1) information that is intended to be read on the screen by the user while using the software (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.15) Syn: on-screen information See also: printed documentation, embedded documentation

**one-address instruction.** (1) computer instruction that contains one address field (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: single-address instruction, single-operand instruction See also: multiaddress instruction, two-address instruction, three-address instruction, four-address instruction, zero-address instruction

**one-ahead addressing.** (1) method of implied addressing in which the operands for a computer instruction are understood to be in the storage locations following the locations of the operands used for the last instruction executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: repetitive addressing

**one-plus-one address instruction.** (1) computer instruction that contains two address fields, the second containing the address of the instruction to be executed next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: two-plus-one address instruction, three-plus-one address instruction, four-plus-one address instruction

**one-time programming (OTP).** (1) method of recording data in ROM which can only be written once (ISO/IEC/IEEE 24765c:2014)

**one-to-many relationship.** (1) relationship between two state classes in which each instance of one class, referred to as the child class, is specifically constrained to relate to no more than one instance of a second class, referred to as the parent class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.131)

**online.** (1) pertaining to a system or mode of operation in which input data enter the computer directly from the point of origin or output data are transmitted directly to the point where they are used (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to a device or process that is under the direct control of the central processing unit of a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) pertaining to the operation of a functional unit when under the control of a computer (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: batch, conversational, real time

**online documentation.** (1) information accessed by the user through the use of software (ISO/IEC/IEEE 24765a:2011) Note: can be context-sensitive

**online help.** (1) information about the software that is intended to be read on the screen by the user while using the software (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.28) Note: Online help can be displayed in a variety of forms (contextual help, screen tips, and examples).

**ontology.** (1) logical structure of the terms used to describe a domain of knowledge, including both the definitions of
the applicable terms and their relationships (ISO/IEC/IEEE 24765:2020)

**OOD.** (1) object-oriented design (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**OPA.** (1) organizational process asset (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**open distributed processing.** (1) distributed processing designed to conform to ODP standards (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.4)

**open subroutine.** (1) subroutine that is copied into a computer program at each place that it is called (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: direct insert subroutine See also: closed subroutine, inline code, macro

**operability.** (1) degree to which a product or system has attributes that make it easy to operate and control (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.4.3) (2) degree to which an IT service has attributes that make it easy to operate and control (ISO/IEC TS 25011:2011 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.2.3) Note: Operability corresponds to controllability, (operator) error tolerance, and conformity with user expectations as defined in ISO 9241-110.

**operable.** (1) state of being able to perform the intended function (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**operand.** (1) variable, constant, or function upon which an operation is to be performed (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**operating environment (software).** (1) set of software operating concurrently on a specified computer system (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.20) Syn: operating environment software

**operating mode.** (1) type of operation for a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**operating system.** (1) collection of software, firmware, and hardware elements that controls the execution of computer programs and provides such services as computer resource allocation, job control, input/output control, and file management in a computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**operation.** (1) in computer mathematics, the action specified by an operator on one or more operands. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) in programming, a defined action that can be performed by a computer system. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) running a computer system in its intended environment to perform its intended functions. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (4) interaction between a client object and a server object which is either an interrogation or an announcement (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.2) (5) property that is a mapping from the (cross product of the) instances of the class and the input argument types to the (cross product of the) instances of the other (output) argument types (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.132) (6) action needed to perform an activity (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.11) (7) arithmetic or logical operation performed in an algorithmic and manipulation BFC (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size
measurement method, 3.7) (8) identifiable entity that denotes the indivisible primitive of service provision that can be requested (ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3.8) Note: An operation can consist of other operations.

operation and maintenance costs. (1) costs associated with using an asset as well as costs of keeping it in a usable condition (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

operation and maintenance phase. (1) period of time in the software life cycle during which a software product is employed in its operational environment, monitored for satisfactory performance, and modified as necessary to correct problems or to respond to changing requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

operation code. (1) character or set of characters that specifies a computer operation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: op code

operation exception. (1) exception that occurs when a program encounters an invalid operation code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: addressing exception, data exception, overflow exception, protection exception, underflow exception

operation field. (1) field of a computer instruction that specifies the operation to be performed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: function field, operation part See also: address field

operation interface. (1) interface in which all the interactions are operations (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.7)

operation interface signature. (1) interface signature for an operation interface (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.12) Note: An operation interface signature comprises a set of announcements and interrogation signatures as appropriate, one for each operation type in the interface, together with an indication of causality (client or server, but not both) for the interface as a whole, with respect to the object which instantiates the template.

operational. (1) pertaining to a system or component that is ready for use in its intended environment (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) pertaining to a system or component that is installed in its intended environment (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) pertaining to the environment in which a system or component is intended to be used (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

operational concept (OpsCon). (1) verbal and graphic statement of an organization's assumptions or intent in regard to an operation or series of operations of a system or a related set of systems (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.28) (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.14) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.25) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.26) Note: The operational concept is designed to give an overall picture of the operations using one or more specific systems, or set of related systems, in the organization's operational environment from the users' and operators' perspective. See also: concept of operations

operational scenario. (1) description of an imagined sequence of events that includes the interaction of the product
or service with its environment and users, as well as interaction among its product or service components (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.15) Note:

Operational scenarios are used to evaluate the requirements and design of the system and to verify and validate the system.

**operational testing.** (1) testing conducted to evaluate a system or component in its operational environment (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: development testing, acceptance testing, qualification testing

**operations.** (1) ongoing execution of activities that produce the same product or provide a repetitive service (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**operator.** (1) entity that performs the operation of a system (2) individual or organization that performs the operations of a system (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.29) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.26) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.27) (3) entity that performs the operations of a system (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.29) (4) mathematical or logical symbol that represents an action to be performed in an operation (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (5) individual or organization that operates the system (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.4) (6) individual or an organization that contributes to the functionality of a system and draws on knowledge, skills, and procedures to contribute the function (INCOSE Systems Engineering Handbook, 4th ed.) (7) symbol representing the name of a function (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.21) Note: The role of operator and the role of user can be vested, simultaneously or sequentially, in the same individual or organization. An individual operator combined with knowledge, skills and procedures can be considered as an element of the system. An operator can perform operations on a system that is operated, or of a system that is operated, depending on whether or not operating instructions are placed within the system boundary. See also: secondary user

**operator manual.** (1) document that provides the information necessary to initiate and operate a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: Typically described are procedures for preparation, operation, monitoring, and recovery. An operator manual is distinguished from a user manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. Syn: operator's manual, operations manual See also: diagnostic manual, installation manual, programmer manual, support manual, user manual

**opportunity.** (1) a risk that would have a positive effect on one or more project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: threat

**opportunity cost.** (1) implicit cost associated with investing money in a certain activity, so that it is no longer available for investing elsewhere (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: Making an investment means that the same money cannot be invested elsewhere, where it could be earning the MARR See also: minimum attractive rate of return (MARR)
opportunity study. (1) study to examine a problem and determine whether or not it requires being solved during the time period under consideration (ISO/IEC 2382:2015 Information technology -- Vocabulary)

OpsCon. (1) operational concept (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2) Syn: OPSCON

optical disc (OD). (1) disk which stores binary data in the form of pits which interrupt the reflection of light from a laser (ISO/IEC/IEEE 24765c:2014)

optimistic duration. (1) estimate of the shortest activity duration that takes into account all of the known variables that could affect performance (ISO/IEC 24765h:2019)

optimization analysis. (1) balance of competing components to achieve the best performance under the situation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For example, an algorithm that runs faster will typically use more memory. Optimization balances the value of a faster run time against the cost of additional memory.

optimizing process. (1) quantitatively managed process that is improved based on an understanding of the common causes of variation inherent in the process. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The focus of an optimizing process is on continually improving the range of process performance through both incremental and innovative improvements.

optional. (1) syntax keyword used to specify a partial mapping (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.133) Note: Optional recommendations are expressed using "should". Syn: O See also: mandatory, partial

optional attribute. (1) attribute that can have no value for an instance (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.134)

optional nonidentifying relationship. (1) nonidentifying relationship in which an instance of the child entity can exist without being related to an instance of the parent entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.135) See also: mandatory nonidentifying relationship. nonidentifying relationship [key style]

optional requirement. (1) requirement of a normative document that must be fulfilled in order to comply with a particular option permitted by that document (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.6) Note: An optional requirement is either: a) one of two or more alternative requirements, or b) an additional requirement that is fulfilled only if applicable and can otherwise be disregarded.

optional task. (1) verification and validation (V&V) task that can be added to the minimum V&V tasks to address specific application requirements (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (2) task that can be added to the minimum testing tasks to address specific requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)


ORB core. (1) ORB component which moves a request from a client to the appropriate adapter for the target object
order clash. (1) in software design, a type of structure clash in which a program must deal with two or more data sets that have been sorted in different orders (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data structure-centered design.

ordinal scale. (1) scale in which the measurement values are rankings (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For example, the assignment of defects to a severity level is a ranking See also: interval scale, nominal scale, ratio scale.

organization. (1) group of people and facilities with an arrangement of responsibilities, authorities and relationships (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.4) (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.30) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.27) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.28) (2) person or group of people that has its own functions with responsibilities, authorities and relationships to achieve its objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management services--Requirements, 3.38) (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.4) (3) person or a group of people and facilities with an arrangement of responsibilities, authorities and relationships to achieve its objectives (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.34) (4) person or a group of people and facilities with an arrangement of responsibilities, authorities, and relationships (INCOSE Systems Engineering Handbook, 4th ed.) Note: An identified part of an organization (even as small as a single individual) or an identified group of organizations can be regarded as an organization if it has responsibilities, authorities and relationships. A body of persons organized for some specific purpose, such as a club, union, corporation, or society, is an organization. An organization can be public or private. The arrangement is generally orderly. Syn: organisation.

organization chain. (1) constellation of organizations that have business relationships with one another (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.25) Note: The following are two types of chains identified:
- supply chains of the IT organizations that are involved in the management and operation of the application (application manager, computer center, workspace manager, network manager, business information manager, suppliers, etc.);
- business chains in which the user organization using the application participates (the business process supported by the application forms part of a chain across several organizations; for example, the chain of criminal justice, the healthcare chain).

organization chart. (1) graphical depiction of hierarchies and interrelationships among persons working together (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

organization level. (1) management level or levels responsible for managing one or more data processing or information systems organizations (ISO/IEC/IEEE 24765a:2011)

organizational breakdown structure (OBS). (1) A hierarchical representation of the project organization that
illustrates the relationship between project activities and the organizational units that will perform those activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

organizational learning. (1) a discipline concerned with the way individuals, groups, and organizations develop knowledge (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

organizational management. (1) task and responsibilities to direct and control an organization (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.1.1) Syn: OM

organizational management profile. (1) profile targeted at very small entities (VSEs) to provide them with additional organizational management guidance and selected requirements (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.1.2)

organizational maturity. (1) extent to which an organization has explicitly and consistently deployed processes that are documented, managed, measured, controlled, and continually improved. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Organizational maturity can be measured via appraisals.

organizational objective. (1) overarching objective that sets the context and direction for an organization's activities (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.39) Note: Organizational objectives are established through the strategic level planning activities of the organization.

organizational plan. (1) documented information that specifies the programs to achieve the organizational objectives (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.40)

organizational policy. (1) guiding principle typically established by senior management that is adopted by an organization to influence and determine decisions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

organizational process assets. (1) plans, processes, policies, procedures and knowledge bases, specific to and used by the performing organization (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) artifacts that relate to describing, implementing, and improving processes, such as policies, measurements, process descriptions, and process implementation support tools (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The term 'process assets' is used to indicate that these artifacts are developed or acquired to meet the business objectives of the organization and that they represent investments by the organization that are expected to provide current and future business value. See also: process asset library

organizational process maturity. (1) extent to which an organizational unit consistently implements processes within a defined scope that contributes to the achievement of its business needs (current or projected) (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.2) Syn: organizational maturity

organizational profile. (1) set of process profiles (ISO/IEC TR 29110-3-1:2015 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 3-1: Assessment Guide, 3.3) Note: Profiles conform to the organizational maturity levels that correspond to the basic, intermediate, and advanced profiles.
organizational risk. (1) risks that inhibit the achievement of business values and product line objectives (ISO/IEC 26556:2018 Information technology-Software and systems engineering-Tools and methods for product line organizational management, 3.1)


organizational test specification. (1) document that provides information about testing for an organization, i.e. information that is not project-specific (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.24)

organizational test strategy. (1) document that expresses the generic requirements for the testing to be performed on all the projects run within the organization, providing detail on how the testing is to be performed (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.25)

organizational unit. (1) part of an organization that is the subject of measurement (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.30) (2) identified part of an organization that deploys one or more processes that operate within a coherent set of business goals and which forms the basis for the scope of an assessment (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.14)

Note: An organizational unit is typically part of a larger organization, although in a small organization the organizational unit can be the whole organization.

origin. (1) address of the initial storage location assigned to a computer program in main memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembled origin, loaded origin, starting address

origin attribute. (1) classification of software as either developed or nondeveloped (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

original equipment manufacturer license. (1) license for products or components that are created or manufactured by one company and licensed by another company (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.17) Syn: OEM license

original source statement. (1) source statement that is obtained from an external product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

orphan page. (1) page on a website with no link from the home page or other page on the website (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.20)


OSF. (1) Open Software Foundation (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

OSS. (1) open source software (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2) See also: FOSS

OT. (1) operational test (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

OTE. (1) operational test and evaluation (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2) Syn: OT&E, OT & E

OTP. (1) one-time programming (ISO/IEC/IEEE 24765c:2014) (2) one-time password (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

OTRR. (1) operational test readiness review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

outer cardinality. (1) number of allowed instances of a participating data object from the viewpoint of the other participants in the relationship (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.6.1) See also: inner cardinality


output arc (of a transition). (1) arc directed from the transition to a place (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.22)

output argument. (1) an argument that has not been specified as an input argument (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.136) Note: It is possible for an output argument to have no value at the time a request is made. See also: input argument

output arrow. (1) arrow or arrow segment that expresses IDEF0 output (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.90) Note: That is, an object type set whose instances are created by a function by transforming the function's input. The arrowtail of an output arrow is attached to the right side of a box.

output assertion. (1) logical expression specifying one or more conditions that program outputs must satisfy in order for the program to be correct (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: input assertion, loop assertion, inductive assertion method

output primitive. (1) primitive that includes source statements, function points, and documents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

output product. (1) the physical form that information can take and that an application distributes (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

outsider's viewpoint. (1) perspective of a potential acquirer who does not own either an existing system nor its proposed replacement, and who evaluates the alternatives of buying the existing system at its salvage value or buying a replacement candidate system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: allows sunk cost and salvage values to be properly accounted for in a decision to replace a system

outsource. (1) make an arrangement in which an external organization performs part of an organization's function or process (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems-Requirements, 3.41)

overall project risk. (1) the effect of uncertainty on the project as a whole, arising from all sources of uncertainty including individual risks, representing the exposure of stakeholders to the implications of variations in project outcome, both positive and negative (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

overfitting. (1) generation of a machine learning model that corresponds too closely to the training data, resulting in a model that finds it difficult to generalize to new data (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.51)

overflow exception. (1) exception that occurs when the result of an arithmetic operation exceeds the size of the storage location designated to receive it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: addressing exception, data exception, operation exception, protection exception, underflow exception

overhead time. (1) amount of time a computer system spends performing tasks that do not contribute directly to the progress of any user task (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

overlay. (1) storage allocation technique in which computer program segments are loaded from auxiliary storage to main storage when needed, overwriting other segments not currently in use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) computer program segment that is maintained in auxiliary storage and loaded into main storage when needed, overwriting other segments not currently in use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to load a computer program segment from auxiliary storage to main storage in such a way that other segments of the program are overwritten (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

overlay supervisor. (1) routine that controls the sequencing and positioning of overlays (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

overload. (1) to assign an operator, identifier, or literal more than one meaning, depending upon the data types associated with it at any given time during program execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
override. (1) the ability of a property in a subclass to re-specify the realization of an inherited property of the same name while retaining the same meaning (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.137)

overriding property. (1) property in a subclass that has the same meaning and signature as a similarly named property in one of its superclasses, but has a different realization (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.138)

OVM. (1) orthogonal variability model (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 4)

owned attribute. (1) attribute of an entity that has not migrated into the entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.139) Note: [key style]

owner. (1) person or organization that owns the copyright for the Candidate FSM method (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.7)

owner of the FSM method. (1) the person or organization that owns the intellectual property rights for the FSM method (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.7)


P-V pair. (1) combination of a test item parameter with a value assigned to that parameter, used as a test condition and coverage item in combinatorial test design techniques (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.22)

P/T net. (1) Place/Transition net (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.2.4)

PA. (1) process attribute (ISO/IEC TR 29110-3-1:2015 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 3-1: Assessment Guide, 4.2)

PaaS. (1) platform as a service (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) See also: SaaS

pack. (1) to store data in a compact form in a storage medium, using known characteristics of the data and medium in such a way as to permit recovery of the data (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: unpack

c package. (1) separately compilable software component consisting of related data types, data objects, and subprograms (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) to combine related components into a single, deployable item (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) (3) namespace for the grouped elements (ISO/IEC 30130:2016 Software engineering --Capabilities of software testing tools) See also: data abstraction, encapsulation, information hiding

packaging. (1) in software development, the assignment of modules to segments to be handled as distinct physical
units for execution by a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

padding. (1) technique of filling out a fixed-length block of data with dummy characters, words, or records (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) dummy characters, words, or records used to fill out a fixed-length block of data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

page. (1) fixed-length segment of data or of a computer program treated as a unit in storage allocation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in a virtual storage system, a fixed-length segment of data or of a computer program that has a virtual address and is transferred as a unit between main and auxiliary storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) screenful of information on a video display terminal (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) structuring mechanism used to split a large net graph into smaller parts, which are also the units of the net to be printed (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.8) See also: paging

page breakage. (1) portion of main storage that is unused when the last page of data or of a computer program does not fill the entire block of storage allocated to it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: paging

page frame. (1) block of main storage having the size of, and used to hold, a page (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: paging

page reference. (1) expression that unambiguously identifies a model page (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.91) Note: The page reference incorporates a diagram reference to the associated diagram, the type of page, and any sequencing data needed to distinguish different pages of the same type that are associated with the same diagram.

page swapping. (1) exchange of pages between main storage and auxiliary storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: paging

page table. (1) table that identifies the location of pages in storage and gives significant attributes of those pages (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: paging


page zero. (1) in the paging method of storage allocation, the first page in a series of pages (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

page-by-page reviewing. (1) technique in which reviewers review a work product in a sequential order (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.11)

pager. (1) routine that initiates and controls the transfer of pages between main and auxiliary storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: paging

paging. (1) storage allocation technique in which programs or data are divided into fixed-length blocks called pages, main storage is divided into blocks of the same length called page frames, and pages are stored in page frames, not necessarily contiguously or in logical order (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) storage allocation technique in which programs or data are divided into fixed-length blocks called pages, main storage is divided into blocks of the same length called page frames, and pages are transferred between main and auxiliary storage allocation by a computer

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.

pair review. (1) informal review of a work product performed by two suitably qualified people other than the author working together (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.12)

pairwise testing. (1) black-box test design technique in which test cases are designed to execute all possible discrete combinations of each pair of input parameters (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.52) Note: Pairwise testing is the most popular form of combinatorial testing.

PAM. (1) process assessment model (ISO/IEC TR 29110-3-1:2015 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 3-1: Assessment Guide, 4.2)

parallel. (1) pertaining to the simultaneous transfer, occurrence, or processing of the individual parts of a whole, such as the bits of a character, using separate facilities for the various parts (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: serial (1), concurrent

parallel classes. (1) pair of classes that are distinct, are not mutually exclusive and have a common generic ancestor class and for which neither is a generic ancestor of the other (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.140)

parallel construct. (1) program construct consisting of two or more procedures that can occur simultaneously (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

parallel run operation. (1) operation of two information processing systems, a given one and its intended replacement, with the same application and source data, for comparison and confidence (ISO/IEC 2382:2015 Information technology -- Vocabulary)

parameter. (1) variable that is given a constant value for a specified application (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) constant, variable, or expression that is used to pass values between software modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) symbol that can take a range of values defined by a set (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.24) (4) parts of the model that are learned from applying the training data to the algorithm (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.53) See also: adaptation

parameterized collection class. (1) collection class restricted to hold only instances of a specified type (class) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.141)

parametric estimating. (1) estimating technique in which an algorithm is used to calculate cost or duration based on historical data and project parameters. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

parent box. (1) ancestral box related to its child diagram by exactly one parent/child relationship (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.93) Note: That is, a box
detailed by a child diagram. The existence of this child diagram is indicated by a box detail reference.

**parent diagram.** (1) diagram that contains a parent box *(IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.94)*

**parent entity.** (1) entity in a specific relationship whose instances can be related to a number of instances of another entity (child entity) *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.142) Note: [key style]*

**parent function.** (1) function modeled by a parent box *(IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.95)*

**Pareto diagram.** (1) histogram, ordered by frequency of occurrence, that shows how many results were generated by each identified cause *(ISO/IEC/IEEE 24765h:2019) Syn: Pareto chart*

**parking lot diagram.** (1) displayed listing of incomplete tasks or user stories not yet being worked or completed. This listing can be grouped by function, with the estimated priority and expected date to start, finish, or dispose of the items. *(Software Extension to the PMBOK(R) Guide Fifth Edition)*

**parse.** (1) to determine the syntactic structure of a language unit by decomposing it into more elementary subunits and establishing the relationships among the subunits *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**parser.** (1) software tool that parses computer programs or other text, often as the first step of assembly, compilation, interpretation, or analysis *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**partial.** (1) incomplete mapping *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.143) Note: That is, some instances map to no related instance. An attribute can be declared partial, meaning it has no value. A participant property is declared optional as part of the relationship syntax. An operation is declared partial when it has no meaning for some instances, i.e., it does not give an answer or produce a response. See also: total, mapping completeness, optional*

**partial cluster.** (1) subclass cluster in which an instance of the superclass exists without also being an instance of any of the subclasses *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.144) Syn: incomplete cluster See also: total cluster, superclass*

**partial correctness.** (1) in proof of correctness, a designation indicating that a program's output assertions follow logically from its input assertions and processing steps *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: total correctness*

**participant property.** (1) property of a state class that reflects that class' knowledge of a relationship in which instances of the class participate *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.145) Note: When a relationship exists between two state classes, each class contains a participant property for that relationship. A participant property is a mapping from a state class to a related (not necessarily distinct) state class. The name of each participant property is the name of the role that the other class plays in the relationship, or it is simply the name of the class at the other end of the relationship (as long as using the class name does not cause ambiguity). A value of a participant property is the identity of a related instance.*

**partition.** (1) set of software functions within an application boundary that share homogeneous criteria and values
A partition requires development effort, which may not be reflected when sizing the functional aspect of the project/product, using FPA. Partition is designed to improve non-functional perspective of the users, such as maintainability, portability, or installability, and is not based on technical or physical considerations. Partitions do not overlap.

**partitioning.** (1) decomposition; the separation of the whole into its parts

**party.** (1) organization entering into an agreement

**pass.** (1) single cycle in the processing of a set of data, usually performing part of an overall process

**pass/fail criteria.** (1) decision rules used to determine whether a software item or a software feature passes or fails a test

**passive I/O device.** (1) device that does not generate an interrupt on completion of an input or output operation

**passive I/O device interface task.** (1) task that interfaces to a passive I/O device and either reads from it or writes to it on demand

**passive object.** (1) object with no thread of control

**patch.** (1) modification made directly to an object program without reassembling or recompiling from the source program

**path.** (1) in software engineering, a sequence of instructions that are performed in the execution of a computer program

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.

Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
sequence of executable statements of a test item (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.23)

path analysis. (1) analysis of a computer program to identify all possible paths through the program, to detect incomplete paths, or to discover portions of the program that are not on any path (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

path condition. (1) set of conditions that must be met in order for a particular program path to be executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

path convergence. (1) a relationship in which a schedule activity has more than one predecessor (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

path divergence. (1) a relationship in which a schedule activity has more than one successor (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

path expression. (1) logical expression indicating the input conditions that must be met in order for a particular program path to be executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

path testing. (1) testing designed to execute all or selected paths through a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: branch testing, statement testing

pathological coupling. (1) type of coupling in which one software module affects or depends upon the internal implementation of another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling

pause. (1) to suspend the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: halt, stop

payoff matrix. (1) in decisions under uncertainty, a table relating the desirability of a set of alternatives to a set of future states (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


PCO. (1) Point of Control and Observation (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)


PDL. (1) program design language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


PDR. (1) preliminary design review (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

peer desk check. (1) informal review where the author and a colleague walk through a work product (ISO/IEC
peer review. (1) review of work products performed by others qualified to do the same work (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.13) Note: often performed during development of the work products to identify defects for removal. The intent is to increase the quality of the work product as well as to reduce cost by fixing defects as soon as possible. See also: buddy check, inspection, structured walkthrough, work product

peer software. (1) piece of software that resides in the same layer as, and exchanges data with, another piece of software (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.21)

PEO. (1) program executive office (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

percent complete. (1) an estimate, expressed as a percent, of the amount of work that has been completed on an activity or a work breakdown structure component (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

perceptual reference point. (1) reference point at which there is some interaction between the system and the physical world (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 15.3.2)

perfective maintenance. (1) modification of a software product after delivery to detect and correct latent faults in the software product before they are manifested as failures (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.42) (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.7) (2) software maintenance performed to improve the performance, maintainability, or other attributes of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) improvements in software's performance or functionality, for example, in response to user suggestions and requests (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Perfective maintenance provides enhancements for users, improvement of program documentation, and recoding to improve software performance, maintainability, or other software attributes. See also: adaptive maintenance, corrective maintenance

perform integrated change control. (1) the process of reviewing all change requests, approving changes, and managing changes to deliverables, organizational process assets, project documents, and the project management plan, and communicating their disposition (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

perform qualitative risk analysis. (1) the process of prioritizing risks for further analysis or action by assessing and combining their probability of occurrence and impact as well as other characteristics (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

perform quantitative risk analysis. (1) the process of numerically analyzing the combined effect of identified individual project risks and other sources of uncertainty on overall project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

performance. (1) degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary (2) extent to which the execution of an application in the production environment achieves its purpose in terms of speed of input, transfer, processing, storage and output (the response speed of an application observed by an end user) (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.26) (3) measurable result (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.42) (4) quantitative measure characterizing a physical or functional attribute relating to the execution of a process, function, activity, or task (INCOSE Systems Engineering Handbook, 4th ed.) Note: Performance attributes include quantity (how many or how much), quality (how well), timeliness (how responsive, how frequent), and readiness (when, under which circumstances). Performance can relate to the management of activities, processes, products (including services), systems, or organizations.

performance analysis. (1) quantitative analysis of a real-time system (or software design) executing on a given hardware configuration with a given external workload applied to it (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

performance baseline. (1) result from a normal execution of a performance workload against a system without performing disturbance injection (ISO/IEC 25045:2010 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation module for recoverability, 4.1)

performance deficiency. (1) difference between the required (or desired) level of performance and the actual performance (ISO/IEC 25064:2013 Systems and software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: User needs report, 4.9) Note: Performance deficiencies can include deficiencies in measured customer satisfaction. Deficiency data is obtainable only in environments where specific performance requirements exist.

performance efficiency. (1) performance relative to the amount of resources used under stated conditions (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.2) Note: Resources can include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, storage media).

performance measurement baseline. (1) integrated scope, schedule, and cost baselines used for comparison to manage, measure, and control project execution (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: PMB

performance metrics. (1) metrics used to evaluate machine learning models that are used for classification (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.54) Syn: performance measures

performance requirement. (1) measurable criterion that identifies a quality attribute of a function or how well a functional requirement shall be accomplished (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) (2) system or software requirement specifying a performance characteristic that a system/software system or system/software component must possess (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) requirement that imposes conditions on a functional requirement (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: A performance requirement is an attribute of a functional requirement. See also: nonfunctional requirement
**performance reviews.** (1) a technique that is used to measure, compare, and analyze actual performance of work in progress on the project against the baseline *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**performance specification.** (1) document that specifies the performance characteristics that a system or component must possess *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)*  
*Note: often part of a requirements specification. These characteristics typically include speed, accuracy, and memory usage.*

**performance testing.** (1) type of testing conducted to evaluate the degree to which a test item accomplishes its designated functions within given constraints of time and other resources *(ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.27)*  
*See also: functional testing*

**performed process.** (1) process that accomplishes the needed work to produce work products *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*  
*Note: satisfies the specific goals of the process area performing organization. (1) an enterprise whose personnel are most directly involved in doing the work of the project or program *(ISO/IEC/IEEE 24765h:2019)*

**periodic I/O device interface task.** (1) task that interfaces to a passive I/O device and polls it regularly *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**periodic task.** (1) task that a timer event activates at regular intervals *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**peripheral equipment.** (1) device that is controlled by and can communicate with a particular computer *(ISO/IEC 2382:2015 Information technology -- Vocabulary)*

**permanence.** (1) degree to which failures can affect object state changes due to completed transactions *(ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.7.1.5)*

**permission.** (1) prescription that a particular behavior is allowed to occur *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.5)*

**perpetual license.** (1) license for a software entitlement granted in perpetuity *(ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.18)*  
*Note: The alternative to a perpetual license is a term or subscription-based license.*

**persistence.** (1) property that an object continues to exist across changes of contractual context or of epoch *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.3.1)*

**persistence schema.** (1) specification of constraints on the use of specific processing, storage and communication functions *(ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 16.5.1.1)*

**persistence transparency.** (1) distribution transparency which masks, from an object, the deactivation and reactivation of other objects (or itself) *(ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.4.1.7)*  
*Note: Deactivation and reactivation are often used to maintain the persistence of an object when a system is unable to provide it with processing, storage and communication functions continuously.*

**persistent.** (1) for a URL, describing a reference that does not need to change at the link in a document, and can still reach the desired object even though that object has changed locations *(ISO/IEC/IEEE 23026:2015 Systems and
persistent storage. (1) storage which enables a functional process to store data beyond the life of the functional process, or which enables a functional process to retrieve data stored by another functional process, or stored by some other process (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.22) Note: As persistent storage is on the software side of the boundary; it is not considered to be a functional user of the software being measured.

persona. (1) model of a user with defined characteristics, based on research (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.29) (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.12) (2) representation of a type of user that includes a concise summary of the characteristics of the user that is most informative to the design or illustrative of specific user requirements (ISO/IEC 25063:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description) Note: A persona typically includes behavior patterns, goals, skills, attitudes, and environment, with a few fictional personal details to make the persona a realistic character.

personal computer (PC). (1) microcomputer primarily intended for stand-alone use by an individual (ISO/IEC 2382:2015 Information technology -- Vocabulary)

personal protective equipment. (1) any device or appliance designed to be worn or held by an individual for protection against one or more health and safety hazards (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.26)

personnel. (1) individual expected to perform duties on behalf of the organization, including officers, employees, and contractors (ISO/IEC/IEEE 24765g:2018)

personnel management. (1) management of activities involving hiring, retaining, promoting, training, and terminating personnel (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

perspective-based reading. (1) form of role-based reviewing that uses checklists and involves the creation of prototype deliverables to check the completeness and other quality characteristics of the work product (ISO/IEC 20246:2017 Software and systems engineering – Work product reviews, 3.15)

PERT. (1) program evaluation review technique (ISO/IEC/IEEE 16326:2019 Systems and software engineering -- Life cycle processes -- Project management, 3)

pessimistic duration. (1) estimate of the longest activity duration that takes into account all of the known variables that could affect performance (ISO/IEC/IEEE 24765h:2019)

PESTEL. (1) political, economic, social, technological, environmental, and legal (ISO/IEC/IEEE 24765f:2016)

Petri net. (1) algebraic structure with two sets, one called places and the other called transitions, together with their associated relations and functions, and named after their inventor, Carl Adam Petri (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.25) Note: A Petri net is usually represented as a graph having two types of nodes (called places and transitions) connected by arcs, and markings (called tokens) indicating dynamic properties. Syn: PN

Petri net with priorities. (1) Petri net having priorities which can be used for selecting the enabled transitions
according to the given priority scheme (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.26)

**Petri net with time.** (1) Petri net having timing constraints associated with the nodes or arcs (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.27)

**PG.** (1) profile group (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 4.2)

**phase.** (1) period of time in the life cycle during which activities are performed that enable achievement of objectives for that phase (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.15)

**phase gate.** (1) a review at the end of a phase in which a decision is made to continue to the next phase, to continue with modification, or to end a project or program (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**physical configuration audit (PCA).** (1) audit conducted to verify that a configuration item, as built, conforms to the technical documentation that defines it (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (2) evaluation to ensure that the operational system or product conforms to the operational and configuration documentation (INCOSE Systems Engineering Handbook, 4th ed.) Note: For software, the purpose of the software physical configuration audit (PCA) is to ensure that the design and reference documentation is consistent with the as-built software product. See also: functional configuration audit

**physical requirement.** (1) requirement that specifies a physical characteristic that a system or system component must possess (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: design requirement, functional requirement, implementation requirement, interface requirement, performance requirement

**physical source statement (PSS).** (1) source statement considered as a line of code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: logical source statement

**Pla.** (1) provide instruments activity (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)


**picture.** (1) illustration that shows the actual appearance of physical objects (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.32)

**PII.** (1) personally identifiable information (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

**pilot project.** (1) project designed to test a preliminary version of an information processing system under actual but limited operating conditions and which will then be used to test the definitive version of the system (ISO/IEC 2382:2015 Information technology -- Vocabulary)


**PIN.** (1) personal identification number (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)
pipeline. (1) software or hardware design technique in which the output of one process serves as input to a second, the output of the second process serves as input to a third, and so on, often with simultaneity within a single cycle time (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)

pixel. (1) smallest element of a screen display; short for 'picture element' (ISO/IEC/IEEE 24765a:2011)


place. (1) node of a net, usually represented by an ellipse (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.28)

place/transition net. (1) Petri Net comprising a net graph with positive integers associated with arcs and an initial marking function which associates a natural number of simple tokens ('black dots') with places (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 3.29) Syn: P/T net, PTNG

plan. (1) information item that presents a systematic course of action for achieving a declared purpose, including when, how, and by whom specific activities are to be performed (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.16)

plan communications management. (1) the process of developing an appropriate approach and plan for project communications based on stakeholder’s information needs and requirements, and available organizational assets (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan cost management. (1) the process of numerically analyzing the combined effect of identified individual project risks and other sources of uncertainty on overall project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan procurement management. (1) the process of documenting project procurement decisions, specifying the approach, and identifying potential sellers (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan quality management. (1) the process of identifying quality requirements and/or standards for the project and its deliverables, and documenting how the project will demonstrate compliance with quality requirements and/or standards (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan resource management. (1) the process of defining how to estimate, acquire, manage, and utilize physical and team resources. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan risk management. (1) the process of defining how to conduct risk management activities for a project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan risk responses. (1) The process of developing options, selecting strategies, and agreeing on actions to address overall project risk exposure, as well as to treat individual project risks. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

plan schedule management. (1) the process of establishing the policies, procedures, and documentation for planning, developing, managing, executing, and controlling the project schedule (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
plan scope management. (1) the process of creating a scope management plan that documents how the project scope will be defined, validated, and controlled. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

plan stakeholder engagement. (1) The process of developing approaches to involve project stakeholders, based on their needs, expectations, interests, and potential impact on the project. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

plan standard. (1) standard that describes the characteristics of a scheme for accomplishing defined objectives or work within specified resources *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

planned process. (1) process that is documented by both a description and a plan *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

Note: The related process description and plan are coordinated, and the plan can include standards, requirements, objectives, resources, and assignments.

planned value (PV). (1) the authorized budget assigned to scheduled work *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* Syn: budgeted cost of work scheduled (BCWS)

planning. (1) activities concerned with the specification of a plan *(ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.9)*

planning horizon. (1) consistent time span used to compare the cost of alternatives *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

planning package. (1) a work breakdown structure component below the control account with known work content but without detailed schedule activities *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* See also: control account

planning process group. (1) those processes required to establish the scope of the project, refine the objectives, and define the course of action required to attain the objectives that the project was undertaken to achieve. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

plant. (1) assembly of different systems on a specific site *(IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.27)*


platform. (1) type of computer or hardware device and/or associated operating system, or a virtual environment, on which software can be installed or run *(2) combination of an operating system and hardware that makes up the operating environment in which a program runs *(ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.30)* *(3) type of computer or hardware device and/or associated operating system, or a virtual environment on which software can be installed or run *(ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.23)*

Note: A platform is distinct from the unique instances of that platform, which are typically referred to as devices or instances. See also: device

platform as a service (PaaS). (1) provision of a complete environment of IT resources, such as programming languages, libraries, services, and tools supported by the service provider *(IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)* Note: The level of

---


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.

Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
control over the service provided to the customer can vary with the service provider. In DevOps, PaaS is automated as part of the DevOps pipeline. See also: SaaS

**platform provider.** (1) organization responsible for the platform (ISO/IEC 19770-2:2015 (corr 2017), Information technology -- Software asset management -- Part 2: Software identification tag, 4.1.18) Note: The platform provider is typically the vendor of the relevant operating system or virtual environment.

**PLCC.** (1) plastic leaded chip carrier (ISO/IEC/IEEE 24765c:2014)

**pleasure.** (1) degree to which a user obtains pleasure from fulfilling personal needs (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.3.3) Note: Personal needs can include needs to acquire new knowledge and skills, to communicate personal identity and to provoke pleasant memories.

**plurality.** (1) decisions made by the largest block in a group, even if a majority is not achieved (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


**PM&P.** (1) parts, materials, and processes (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**PMB.** (1) performance measurement baseline (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


**PMI.** (1) Project Management Institute (ISO/IEC/IEEE 24765i:2020)


**PNG.** (1) Portable Network Graphics (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)


**PNML Core Model.** (1) metamodel defining the basic concepts and structure of net graph models that are common to all versions of Petri nets (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.9)

**PNML document.** (1) XML document that contains one or more net graphs (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.10) Syn: Petri Net document
PNML high-level net document. (1) PNML Document that contains one or more net graphs, where all net graphs
conform to high-level Petri nets (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2:
Transfer format, 4.1.11)

PNML symmetric net document. (1) PNML Document that contains one or more net graphs, where all net
graphs conform to symmetric nets (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets --
Part 2: Transfer format)

PO. (1) purchase order (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very
Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group:
Basic profile, 4.2)

POA. (1) portable object adapter (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common
Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.3)

point. (1) measure of vertical distance; there are approximately 28 points to the millimeter (72 points to the inch)
(ISO/IEC/IEEE 24765a:2011)

point design. (1) selection of one design that satisfies the requirements without examining other, potentially more
effective, designs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

pointer. (1) data item that specifies the location of another data item (ISO/IEC/IEEE 24765:2017 Systems and software
engineering-Vocabulary)

policy. (1) clear and measurable statements of preferred direction and behavior to condition the decisions made within
an organization (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items
(documentation), 5.17) (2) intentions and direction of an organization, as formally expressed by its top management
(ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--
Requirements, 3.43) (3) constraint on a system specification foreseen at design time, but whose detail is determined
subsequent to the original design, and capable of being modified from time to time in order to manage the system in
changing circumstances (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference
Model: Foundations, 11.2.8) (4) a structured pattern of actions adopted by an organization such that the organization's
policy can be explained as a set of basic principles that govern the organization's conduct (A Guide to the Project
Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: A rule can be expressed as an obligation, an
authorization, a permission, or a prohibition. Not every policy is a constraint. Some policies represent an empowerment.
Direction includes mandates set by the organization.

policy declaration. (1) element in a specification defined in order to allow incorporation of future constraints,
together with rules determining the allowed form of acceptable constraints and the circumstances in which such
constraints can be applied (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference
Model: Foundations, 11.2.9)

policy envelope. (1) set of acceptable policy values that could be applied at a particular policy declaration (ISO/IEC

policy value. (1) specific constraints associated with a policy in some particular epoch (ISO/IEC 10746-2:2009
Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.11)
policy-setting behavior. (1) behavior defined in a specification via which a policy can be changed (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.12) Syn: policy setting behavior

pop-up. (1) embedded, context-sensitive information that is displayed when invoked by user action (ISO/IEC/IEEE 24765g:2018)

port. (1) surface feature through which clients and other elements of an application environment can interact with a component (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

port alias. (1) replacement relationship in a build specification that identifies a port of one unit with a port of a sub-unit and indicates that interactions at the two ports can be paired identically or compatibly (ISO/IEC/IEEE 24765i:2020) Note: A port alias is used to indicate identity or compatibility of two ports at different levels of assembly in which the port in a lower level unit of assembly serves as a realization of a port at a higher-level unit of assembly.

port-to-port time. (1) elapsed time between the application of a stimulus to an input interface and the appearance of the response at an output interface (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: response time, think time, turnaround time

portability. (1) ease with which a system or component can be transferred from one hardware or software environment to another (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) capability of a program to be executed on various types of data processing systems without converting the program to a different language and with little or no modification (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) degree of effectiveness and efficiency with which a system, product, or component can be transferred from one hardware, software or other operational or usage environment to another (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.8) (4) property that the reference points of an object allow it to be adapted to a variety of configurations (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 15.4.1) Syn: transportability See also: machine-independent

portability testing. (1) type of testing conducted to evaluate the ease with which a test item can be transferred from one hardware or software environment to another, including the level of modification needed for it to be executed in various types of environments (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.28)

portable computer. (1) microcomputer that can be hand-carried for use in more than one location (ISO/IEC 2382:2015 Information technology -- Vocabulary)

portfolio. (1) projects, programs, subportfolios, and operations managed as a group to achieve strategic objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

portfolio management. (1) the centralized management of one or more portfolios to achieve strategic objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) centralized management of one or more portfolios to achieve business objectives (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)
post-closure activities. (1) activities that occur after a software system has been formally accepted by its customer (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: These activities include, but are not limited to, lessons-learned reviews and archiving project materials.

post-compile time. (1) collective name for link time and load time that are right after the compilation of components (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.9)

post-probe. (1) optional phase to prepare action plans for addressing challenge (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.3)

postcondition. (1) condition that is guaranteed to be true after a successful property request (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.147) (2) constraint that must be true when a use case has ended (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) predicate that a specification requires to be true immediately after the occurrence of an action (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.30) Syn: post-condition

postmortem dump. (1) dump that is produced upon abnormal termination of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: post-mortem dump See also: change dump, dynamic dump, memory dump, selective dump, snapshot dump, static dump

postprocessor. (1) computer program or routine that carries out some final processing step after the completion of the primary process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: post-processor See also: preprocessor

power-down mode. (1) energy-saving operational state for a microcontroller unit (MCU) (ISO/IEC/IEEE 24765e:2015)

powertype. (1) type, the instances of which are subtypes of another type called the 'partitioned type' (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 3.12)


PPM. (1) project portfolio management (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecyle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.2)

PPP. (1) program protection plan (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

PPSL. (1) program parts selection list (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)


practice. (1) specific type of activity that contributes to the execution of a process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.8) (2) a specific type of professional or management
activity that contributes to the execution of a process and that may employ one or more techniques and tools (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: conventions, standards practitioner (PT). (1) person or team performing the activities within one or more process areas (ISO/IEC TR 29110-5:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5: Service delivery guidelines, 3.21)

pre-probe. (1) phase to understand an organization's basic context such as current structure, terminology, product maturity level, implementation, and documentation (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.4)

pre-processing. (1) part of the machine learning (ML) workflow that transforms raw data into a state ready for use by the ML algorithm to create the ML model (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.57) Note: Pre-processing can include analysis, normalization, filtering, reformattting, imputation, removal of outliers and duplicates, and evaluating the completeness of the dataset.

precautionary principle. (1) principle that the introduction of a new product or process whose ultimate effects are unknown or disputed should be resisted until such risks are appropriately mitigated (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: The precautionary principle denotes a duty to prevent harm when it is within one's power to do so, even when all the evidence is not in.

precedence diagramming method (PDM). (1) a technique used for constructing a schedule model in which activities are represented by nodes and are graphically linked by one or more logical relationships to show the sequence in which the activities are to be performed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: activity-on-node (AON)

precedence relationship. (1) a logical dependency used in the precedence diagramming method (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: logical relationship precision. (1) degree of exactness or discrimination with which a quantity is stated (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) performance metric used to evaluate a classifier, which measures the proportion of predicted positives that were correct (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.55) See also: accuracy

precompiler. (1) computer program or routine that processes source code and generates equivalent code that is acceptable to a compiler (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: for example, a routine that converts structured FORTRAN to ANSI-standard FORTRAN See also: preprocessor

precondition. (1) condition that is required to be true before making a property request (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.148) (2) constraint that must be true when a use case is invoked (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) predicate that a specification requires to be true for an action to occur (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview, 9.29) Syn: pre-condition

**predicate.** (1) logical expression which evaluates to TRUE or FALSE, normally to direct the execution path in code (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.24)

**predicate data use.** (1) data use associated with the decision outcome of the predicate portion of a decision statement (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.25) Syn: p-use

**prediction.** (1) machine learning function that results in a predicted target value for a given input (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.56)

**predictive action.** (1) action to monitor the condition of an asset and predict the need for preventive action or corrective action (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.44) Syn: condition monitoring, performance monitoring

**predictive life cycle.** (1) a form of project life cycle in which the project scope, time, and cost are determined in the early phase of the life cycle (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**preliminary design.** (1) process of analyzing design alternatives and defining the architecture, components, interfaces, and timing and sizing estimates for a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) result of the process in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: detailed design

**preliminary design review (PDR).** (1) review conducted to evaluate the progress, technical adequacy, and risk resolution of the selected design approach for one or more configuration items; to determine each design's compatibility with the requirements for the configuration item; to evaluate the degree of definition and assess the technical risk associated with the selected manufacturing methods and processes; to establish the existence and compatibility of the physical and functional interfaces among the configuration items and other items of equipment, facilities, software and personnel; and, as applicable, to evaluate the preliminary operational and support documents (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) review as in (1) of any hardware or software component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: critical design review, system design review

**preparation time.** (1) time which elapses before the task submission (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.12) Note: The event of starting the preparation time depends on the definition of the task mode of the following task. See also: task mode

**preprocessor.** (1) computer program or routine that carries out some processing step prior to the primary process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: postprocessor

**prescription.** (1) action that establishes a rule (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.3)

**present worth.** (1) representation of a cash flow as a single instance at the beginning of the planning horizon (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: future worth, annual equivalent

**presentable.** (1) retrievable and viewable (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.18)
presentation device. (1) device used to present data to the intended user of a system (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.30)

prestore. (1) to store data that are required by a computer program or routine before the program or routine is entered (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

prettyprinting. (1) use of indentation, blank lines, and other visual cues to show the logical structure of a program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

preventive action. (1) an intentional activity that ensures the future performance of the project work is aligned with the project management plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) action to eliminate the cause of a potential nonconformity or other undesirable potential situation (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.45) Note: Preventive action is taken to prevent occurrence and to preserve an asset's function, whereas corrective action is taken to prevent recurrence. Preventive action is normally carried out while the asset is functionally available and operable or prior to the initiation of functional failure. Preventive action includes the replenishment of consumables where the consumption is a functional requirement. See also: corrective action

preventive maintenance. (1) modification of a software product after delivery to detect and correct latent faults in the software product before they become operational faults (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.8) (2) maintenance performed for the purpose of preventing problems before they occur (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) designing a software system that is easy to maintain (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) continuously upgrading a system to enable it to cope with current and future changes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

previously developed software. (1) software that has been produced prior to or independent of the project for which the plan is prepared, including software that is obtained or purchased from outside sources (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.2) (2) software that has been produced prior to or independent of the project for which the Plan is prepared, including software that is obtained or purchased from outside sources (IEEE Std 1228-1994 IEEE Standard for Software Safety Plans, 3.1.2)

primary data. (1) data that is collected directly from a subject (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)

primary Ent. (1) entitlement schema (Ent) which encapsulates basic information about an entitlement (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.19) Note: Primary Ents have an <entType> of Initial, Consolidation, AllocationReceived, or TransferReceived. These are base Ents which allow for initial population of an Ent into an Ent library (with the exception of Consolidation, which can be used to replace several previous Ents if desired). Other Ents (called supplemental Ents) can extend the data in these base type Ents. Syn: primary entitlement schema

primary entity-type. (1) in Mk II FPA, one of the main entity-types which has the attributes that the application has been designed to process and/or store (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis --


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition long as the statement " Copyright©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
primary information structure. (1) information structure to which supplemental information structures may be linked (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.24) Syn: primary info struct

primary intent. (1) intent that is first in importance (ISO/IEC 20926:2009 Software and systems engineering--Software measurement--IFPUG functional size measurement method 2009, 3.43)

primary key. (1) candidate key selected as the unique identifier of an entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.149) (2) value that uniquely identifies component instances within the scope of the home that manages them (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: [key style]

primary user. (1) user who interacts with the system to achieve the primary goals (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.11)

primitive. (1) lowest level for which data is collected (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Primitives are directly measurable or countable, or can be given a constant value or condition for a specific measure.

principal. (1) party that has delegated (authority, a function, etc.) to another (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.6.9)


printed documentation. (1) documentation that is either provided in printed form, or provided in electronic form for the customer or user to print (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.33) See also: embedded documentation

priority ceiling protocol. (1) algorithm that provides bounded priority inversion (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: that is, at most one lower-priority task can block a higher-priority task

priority interrupt. (1) interrupt performed to permit execution of a process that has a higher priority than the process currently executing (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

priority inversion. (1) case where a task's execution is delayed because a lower priority task is blocking it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

private. (1) responsibility that is visible only to the class or the receiving instance of the class (available only within methods of the class) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.150) (2) known only within a single routine or module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: protected, public, hidden
private type. (1) data type whose structure and possible values are defined but are not revealed to the user of the type (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: information hiding

privileged instruction. (1) computer instruction that can be executed only by a supervisory program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

PRM. (1) process reference model (ISO/IEC TR 29110-3-1:2015 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 3-1: Assessment Guide, 4.2)

proactive approach. (1) approach of developing an innovative product line or product variations based on organizational predictions that anticipate a stated product need (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.15)

probabilistic system. (1) system whose behavior is described in terms of probabilities, such that its outputs cannot be perfectly predicted (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.58)

probability. (1) extent to which an event is likely to occur (ISO/IEC/IEEE 24765j:2021) (2) mathematically, a real number in the scale 0 to 1 attached to a random event, related to a long-run relative frequency of occurrence or to a degree of belief that an event will occur (ISO/IEC/IEEE 24765j:2021) Note: For a high degree of belief, the probability is near 1. Frequency rather than probability can be used in describing risk. Degrees of belief about probability can be chosen as classes or ranks, such as rare/ unlikely/ moderate/ likely/ almost certain, or incredible/ improbable/ remote/ occasional/ probable/ frequent.

probability and impact matrix. (1) a grid for mapping the probability of each risk occurrence and its impact on project objectives if that risk occurs (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

probe. (1) diagnostic process for investigating the organization's readiness to adopt, or ability to succeed with, product line engineering and management (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.7) Syn: product line technical probe, technical probe

cause is not usually known at the time a problem record is created and the problem management process is responsible for further investigation.

A problem might concern a service or product or a process (step) or any other element of the application management organization.

**problem definition.** (1) statement of a problem, which can include a description of the data, the method, the procedures, and algorithms used to solve it (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: problem description

**problem report (PR).** (1) document used to identify and describe problems detected in a product (IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.9) Note: PRs are either submitted directly to denote faults or established after impact analysis is performed on Modification Requests and faults are found.

**problem state.** (1) in the operation of a computer system, a state in which programs other than the supervisory program can execute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: user state See also: supervisor state

**problem-oriented language.** (1) programming language designed for the solution of a given class of problems (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**procedural cohesion.** (1) type of cohesion in which the tasks performed by a software module all contribute to a given program procedure, such as an iteration or decision process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: coincidental cohesion, communicational cohesion, functional cohesion, logical cohesion, sequential cohesion, temporal cohesion

**procedural language.** (1) programming language in which the user states a specific set of instructions that the computer must perform in a given sequence (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: All widely-used programming languages are of this type. Syn: procedure-oriented language See also: nonprocedural language, algebraic language, algorithmic language, list processing language, logic programming language

**procedural programming language.** (1) computer programming language used to express the sequence of operations to be performed by a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: nonprocedural programming language

**procedure.** (1) information item that presents an ordered series of steps to perform a process, activity, or task (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.19) (2) portion of a computer program that is named and that performs a specific action (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) routine that does not return a value (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) an established method of accomplishing a consistent performance or result. A procedure typically can be described as the sequence of steps that will be used to execute a process. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (5) specified way to carry out an activity or process (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.25) (6) ordered series of steps that a user follows to do one or more tasks (ISO/IEC/IEEE 26513:2017 Systems and...
A procedure defines an established and approved way or mode of conducting business in an organization. It details permissible or recommended methods in order to achieve technical or managerial goals or outcomes. When a procedure is specified as an outcome, the resulting deliverable will typically specify what must be done, by whom, and in what sequence. This is a more detailed level of specification than for a process. Procedures can be documented or not.

**procedure testing. (1)** type of functional suitability testing conducted to evaluate whether procedural instructions for interacting with a test item or using its outputs meet user requirements and support the purpose of their use (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.29)

**process. (1)** set of interrelated or interacting activities which transforms inputs into outputs, (2) set of interrelated or interacting activities which transforms inputs into outputs (3) set of interrelated or interacting activities that transforms inputs into outputs (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.33) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.30) (4) predetermined course of events defined by its purpose or by its effect, achieved under given conditions (ISO/IEC 2382:2015 Information technology -- Vocabulary) (5) to perform operations on data (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (6) collection of steps taking place in a prescribed manner (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.6) (7) in data processing, the predetermined course of events that occur during the execution of all or part of a program (ISO/IEC 2382:2015 Information technology -- Vocabulary) (8) executable unit managed by an operating system scheduler (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (9) system of activities, which use resources to transform inputs into outputs (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.22) (10) a systematic series of activities directed towards causing an end result such that one or more inputs will be acted upon to create one or more outputs (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (11) set of interrelated or interacting activities that use inputs to deliver an intended result (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.36) (12) set of interrelated or interacting activities which transforms inputs into outputs to deliver an intended result (ISO/IEC TR 29110-5:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.23) Note: The term "activities" covers use of resources. A process can have multiple starting points and multiple end points. The prescribed manner can be a partially ordered sequence. A process specification can be a workflow specification. An enterprise specification can define types of processes and process templates. A process can be viewed as a specific instantiation of life cycle processes, adapted within a life cycle model, to create the service or product for the specific requirements and context of a project. When a process definition is specified as an outcome, the resulting deliverable typically specifies inputs and outputs, and gives a general description of expected activities. However, it does not include the same level of detail as for a procedure.

**process action plan. (1)** plan, usually resulting from appraisals, that documents how specific improvements targeting the weaknesses uncovered by an appraisal will be implemented (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.23)
process action team. (1) team that has the responsibility to develop and implement process improvement activities for an organization as documented in a process action plan (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

process architect. (1) person or group that has primary responsibility for creating and maintaining the software life cycle process (SLCP) (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

process architecture. (1) ordering, interfaces, interdependencies, and other relationships among the process elements in a standard process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Process architecture also describes the interfaces, interdependencies, and other relationships between process elements and external processes, such as contract management.

process area. (1) cluster of related practices in an area that, when implemented collectively, satisfies a set of goals considered important for making improvement in that area (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

process assessment. (1) disciplined evaluation of an organizational unit's processes against a process assessment model (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.15) (2) determination of the extent to which the organization's standard processes contribute to the achievement of its business goals and help the organization focus on the need for continuous process improvement (ISO/IEC/IEEE 24765d:2015) process assessment model (PAM). (1) model suitable for the purpose of assessing a specified process quality characteristic, based on one or more process reference models (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.38) (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.9) Note: Process assessment models addressing a specific process quality characteristic can include the identification of the characteristic in the title; for example, a process assessment model addressing process capability can be termed a "process capability assessment model." See also: process reference model (PRM)

process asset library. (1) collection of information that can be useful to those who are defining, implementing, and managing processes in the organization. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: process-related documentation such as policies, defined processes, checklists, lessons-learned documents, templates, standards, procedures, plans, and training materials.

process attribute (PA). (1) measurable property of a process quality characteristic (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.3) Syn: process quality attribute

process attribute outcome. (1) observable result of achievement of a specified process attribute (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.4)

process attribute rating. (1) judgment of the degree of achievement of the process attribute for the assessed process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.5)

process capability. (1) characterization of the ability of a process to meet current or projected business goals (ISO/IEC 33020:2019 Information technology--Process assessment--Process measurement framework for assessment of process capability, 3.4) (2) range of expected results that can be achieved by following a process (ISO/IEC/IEEE
process capability level. (1) characterization of a process on an ordinal measurement scale of process capability (ISO/IEC 33020:2019 Information technology--Process assessment--Process measurement framework for assessment of process capability, 3.5) (2) point on the six-point ordinal scale of process capability that represents the capability of the process (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.39) Note: Each level builds on the capability of the level below.

process component. (1) CORBA component with persistent state, which is not visible to the client, persistent identity, and behavior, which can be transactional (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

process definition. (1) identification of a sequence of steps involving activities, constraints, and resources that are performed for a given purpose (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

process description. (1) documented expression of a set of activities performed to achieve a given purpose (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A process description provides an operational definition of the major components of a process. The description specifies, in a complete, precise, and verifiable manner, the purpose, outcomes, activities and tasks, requirements, design, behavior, or other characteristics of a process. It also can refer to procedures for determining whether these provisions have been satisfied. Process descriptions can be applicable at the project or organizational level.

process dimension. (1) set of process elements in a process assessment model explicitly related to the processes defined in the relevant process reference model(s) (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.10) Note: The elements of the process dimension include processes, process purpose statements, process outcomes, and process performance indicators.

process group. (1) collection of related processes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) team of specialists who facilitate the definition, maintenance, and improvement of processes used by the organization (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

process improvement. (1) actions taken to improve the quality of the organization’s processes aligned with the business needs and the needs of other concerned parties (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.1.7) (2) result of activities that better the performance and maturity of the organization's processes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) actions taken to improve the quality of the organization's processes aligned with the business needs (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.41)

process improvement objective. (1) set of target characteristics established to guide the effort to improve an existing process in a specific, measurable way, either in terms of resultant product or service characteristics, such as quality, performance, and conformance to standards, or in the way in which the process is executed, such as elimination of redundant process steps, combination of process steps, and improvement of cycle time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

process improvement plan. (1) subsidiary plan of the project management plan that details the steps for analyzing processes to identify activities that enhance their value (ISO/IEC/IEEE 24765h:2019)
process improvement support element. (1) way that an organization expresses support for process improvement projects or initiatives (ISO/IEC TR 33014:2013 Information technology--Process assessment--Guide for process improvement, 3.2)

process infrastructure. (1) internal structure of the software life-cycle process, to include lifecycle phases, documentation, baselines, reviews, and products (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)


process management. (1) direction, control, and coordination of work performed to develop a product or perform a service (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

process maturity. (1) extent to which an organizational unit consistently implements processes within a defined scope that contribute to the achievement of its business needs (current or projected) (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.23)

process measurement framework. (1) schema for use in characterizing a process quality characteristic of an implemented process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.6)

process outcome. (1) observable result of the successful achievement of the process purpose (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.34) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.32) Note: An outcome is an artifact, a significant change in state or the meeting of specified constraints. An outcome statement describes one of the following: production of an artifact; a significant change in state; meeting of specified constraints, e.g., requirements, goals.

process owner. (1) person (or team) responsible for defining and maintaining a process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: At the organizational level, the process owner is the person (or team) responsible for the description of a standard process; at the project level, the process owner is the person (or team) responsible for the description of the defined process. A process can therefore have multiple owners at different levels of responsibility.

process performance. (1) extent to which the execution of a process achieves its purpose (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.7)

process performance indicator. (1) assessment indicator that supports the judgment of the process performance of a specific process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.12)

process profile. (1) set of process attribute ratings for an assessed process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.18)

process purpose. (1) high-level objective of performing the process and the likely outcomes of effective

process quality. (1) ability of a process to satisfy stated and implied stakeholder needs when used in a specified context (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.8)

process quality characteristic. (1) measurable aspect of process quality (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.9) (2) category of process attributes that are significant to process quality (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.9)

process quality determination. (1) systematic assessment and analysis of selected processes against a target process profile (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.9)

process quality dimension. (1) set of elements in a process assessment model explicitly related to the process measurement framework for the specified process quality characteristic (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.14)

process quality indicator. (1) assessment indicator that supports the judgment of the process quality characteristic of a specific process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.15)

process quality level. (1) point on a scale of achievement of a process quality characteristic derived from the process attribute ratings for an assessed process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.4.10) (2) representation of the achieved level of a process quality characteristic derived from the process attribute ratings for an assessed process (ISO/IEC TR 29110-3-1:2015 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 3-1: Assessment Guide, 3.2)

process reference model (PRM). (1) model comprising definitions of processes in a domain of application described in terms of purpose and outcomes, together with an architecture describing the relationships between the processes (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.3.16) (2) model comprising definitions of processes in a lifecycle described in terms of purpose and outcomes, together with an architecture describing the relationships between the processes (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.44) See also: process assessment model (PAM)

process risk determination. (1) systematic assessment and analysis of selected processes against a target process profile, carried out with the aim of identifying process-related risks to meet a particular specified requirement (ISO/IEC TR 33015:2019 Information technology--Process assessment--Guidance for process risk determination, 3.1)

process standard. (1) standard that deals with the series of actions or operations used in making or achieving a product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

process tailoring. (1) making, altering, or adapting a process description for a particular end (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For example,

process view. (1) description of how a specified purpose and set of outcomes can be achieved by employing the

**process-related risk. (1)** risk resulting from weaknesses in the performance, management, or deployment of a process (ISO/IEC TR 33015:2019 Information technology--Process assessment--Guidance for process risk determination, 3.2)

**processing logic. (1)** any of the requirements specifically requested by the user to complete an elementary process, such as validations, algorithms, or calculations, and reading or maintaining a file (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009)

**processor. (1)** in a computer, a functional unit that interprets and executes instructions (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: A processor consists of at least an instruction control unit and an arithmetic and logic unit

**procurement audit. (1)** the review of contracts and contracting processes for completeness, accuracy and effectiveness (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**procurement documents. (1)** the documents utilized in bid and proposal activities, which include the buyer’s Invitation for Bid, Invitation for Negotiations, Request for Information, Request for Quotation, Request for Proposal and seller’s responses (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**procurement management plan. (1)** a component of the project or program management plan that describes how a project team will acquire goods and services from outside the performing organization (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**procurement performance review. (1)** structured review of the supplier’s progress to deliver project scope and quality, within cost and on schedule, as compared to the agreement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**procurement statement of work. (1)** describes the procurement item in sufficient detail to allow prospective sellers to determine if they are capable of providing the products, services, or results (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**procurement strategy. (1)** the approach by the buyer to determine the project delivery method and the type of legally binding agreement(s) that should be used to deliver the desired results (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**producer object. (1)** object which is the source of the information conveyed (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.4.3)

**product. (1)** an artifact that is produced, is quantifiable, and can be either an end item in itself or a component item (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) part of the equipment (hardware, software and materials) for which usability is to be specified or evaluated (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.9) (3) result of a process (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.36) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.32) (ISO/IEC/IEEE 24748-1:2018)
output of the software development activities (e.g., document, code, or model) (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) intended or accomplished result of labor, or of a natural or artificial process (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.28) artifact that is produced, is quantifiable and is deliverable to the user as either an end item in itself or a component item (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.12) Note: Generic product categories are hardware (e.g., engine mechanical part); software (e.g., computer program); services (e.g., transport); and processed materials (e.g., lubricant). Hardware and processed materials are generally tangible products, while software or services are generally intangible. Most products comprise elements belonging to different generic product categories. Whether the product is then called hardware, processed material, software, or service depends on the dominant element. Results could be components, systems, software, services, rules, documents, or many other items.

The result could in some cases be many related individual results. See also: activity, deliverable, result

product analysis. (1) process of evaluating a product by manual or automated means to determine if the product has certain characteristics (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) For projects that have a product as a deliverable, it is a tool to define scope that generally means asking questions about a product and forming answers to describe the use, characteristics and other the relevant aspects of what is going to be manufactured. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

product authority. (1) person or persons with overall responsibility for the capabilities and quality of a product (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.37)

product baseline. (1) description of the detailed design at a specific point in time, for production, fielding/deployment, and operations and support (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1) Syn: product configuration baseline, production configuration See also: allocated baseline, developmental configuration, functional baseline, product configuration identification

product configuration identification. (1) current approved or conditionally approved technical documentation defining a configuration item during the production, operation, maintenance, and logistic support phases of its life cycle (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: 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. See also: allocated configuration identification, functional configuration identification, product baseline

product description. (1) document stating properties of software, with the main purpose of helping potential acquirers in the evaluation of the suitability for themselves of the software before purchasing it (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.44) Note: The product description is not a specification; it serves a different purpose.

product engineering. (1) technical processes to define, design, and construct or assemble a product (ISO/IEC/IEEE
product identification. (1) software product name, version, variant, and date information (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.12)

product life cycle. (1) the series of phases that represent the evolution of a product, from concept through delivery, growth, maturity, to retirement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

product line. (1) group of products or services sharing a common, managed set of features that satisfy specific needs of a selected market or mission (INCOSE Systems Engineering Handbook, 4th ed.) (2) set of products or services sharing explicitly defined and managed common and variable features and relying on the same domain architecture to meet the common and variable needs of specific markets (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.16) (3) paradigm for the creation, exploitation, and management of a common platform for a family of products (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.9) Note: Typical goals of product lines are to lower costs, reduce time to market, and improve quality. Syn: product family, software and systems product line. SSPL

product line adoption plan. (1) plan that describes the changes in process, organization structure, and product building methods to get from the current to product line engineering (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.5)

product line adoption scenario. (1) scenario that gives concrete sequence of actions related to product line adoption (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.6)

product line architecture. (1) architecture, including both domain architecture and application architecture (ISO/IEC 26552:2019 Software and systems engineering--Tools and methods for product line architecture design, 3.8)

product line institutionalization. (1) considering product line engineering as part of working culture by involved managers and staff members (ISO/IEC 26562:2019 Software and systems engineering--Methods and tools for product line transition management, 3.2)

product line platform. (1) product line architecture, a configuration management plan, and domain assets, enabling application engineering to effectively reuse and produce a set of derivative products (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.18)

product line reference model. (1) abstract representation of the domain and application engineering life cycle processes; the roles and relationships of the processes; and the assets produced, managed, and used during product line engineering and management (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.19)

product line scoping. (1) process for defining the member products that will be produced within a product line and the major common and variable features among the products, analyzes the products from an economic point of view, and controls and schedules the development, production, and marketing of the product line and its products (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.20)

product line test strategy. (1) scope of domain testing and application testing (ISO/IEC 26554:2018 Information
Many variants may not be implemented in domain engineering. To cope with the impacts of un-implemented variants to domain integration testing and system testing, the appropriate test strategy should be established.

**product management.** (1) definition, coordination, and control of the characteristics of a product during its development cycle *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**product quality.** (1) degree to which the product satisfies stated and implied needs when used under specified conditions *(ISO/IEC 25041:2012 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation guide for developers, acquirers and independent evaluators, 4.11)* *(2)* capability of a system and/or software to satisfy stated and implied needs when used under specified conditions *(ISO/IEC 25020:2019 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Quality measurement framework, 3.17)* Note: This definition differs from the ISO 9000:2015 quality definition, because this definition refers to the satisfaction of stated and implied needs, while the ISO 9000 quality definition refers to the satisfaction of requirements. Syn: system and software product quality

**product requirement.** (1) refinement of customer requirements into the developers’ language, making implicit requirements into explicit derived requirements *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**product risk.** (1) risk that a product could be defective in some specific aspect of its function, quality, or structure *(ISO/IEC/IEEE 29119-3:2013 Software and systems engineering--Software testing--Part 3: Test documentation, 4.8)*

**product roadmap.** (1) timeline with high-level milestones for a product life cycle, particularly the timeline for productive deployment of the product *(ISO/IEC 26556:2018 Information technology-Software and systems engineering-Tools and methods for product line organizational management, 3.2)* *(2)* schedule when the products have to be ready for market launch *(ISO/IEC 26560:2019 Software and systems engineering -- Tools and methods for product line product management, 3.5)* See also: technology roadmap

**product safety label.** (1) label on a product that informs of one or more potential hazards and describes the safety precautions or actions required to avoid the hazard(s) *(IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.29)*

**product scope.** (1) the features and functions that characterize a product, service or result. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**product scope description.** (1) the documented narrative description of the product scope. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**product scoping.** (1) subprocess of product line scoping that determines the product roadmap, that is 1) the targeted markets; 2) the product categories that the product line organization should be developing, producing, marketing, and selling; 3) the common and variable features that the products should provide in order to reach the long and short term business objectives of the product line organization, and 4) the schedule for introducing products to markets *(ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.20)*

**product specification.** (1) document that specifies the design that production copies of a system or component must implement *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* document that describes
the characteristics of a planned or existing product for consideration by potential customers or users (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For software, this document describes the as-built version of the software. See also: design description

**product standard.** (1) standard that defines what constitutes completeness and acceptability of items that are used or produced, formally or informally, during the software engineering process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**product support.** (1) providing of information, assistance, and training to install and make software operational in its intended environment and to distribute improved capabilities to users (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**production.** (1) stage in the life cycle when completed software or documentation is prepared, published, and packaged for distribution (ISO/IEC/IEEE 24765a:2011)

**production library.** (1) software library containing software approved for current operational use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software development library, software repository, system library

**production plan.** (1) description of how domain assets are to be used to develop member products in a product line (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.17)

**production rate.** (1) a measure of the amount of work completed per unit of time, such as user stories or features per week (Software Extension to the PMBOK(R) Guide Fifth Edition) See also: burndown rate, velocity

**productivity.** (1) ratio of work product to work effort (ISO/IEC/IEEE 24765a:2011)

**professional standard.** (1) standard that identifies a profession as a discipline and distinguishes it from other professions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**professionalism.** (1) degree to which the content of the IT service is based on appropriate education, skill, expertise and qualification (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.5.2)

**profile.** (1) set of one or more base standards or profiles and, where applicable, the identification of chosen classes, conforming subsets, options and parameters of those base standards or standardized profiles necessary to accomplish a particular function (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.23) Note: [ISO/IEC TR 10000-1]

**profile group.** (1) collection of profiles which are related by composition of processes (i.e., activities, tasks) or by capability level, or both (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.24) (2) collection of profiles which are related by composition of processes (i.e., activities, tasks), or by requirements sharing or composition (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.45) Syn: PG

**program.** (1) to write a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to design, write, modify, and test programs (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) related projects, subprograms and program activities managed in a coordinated way to obtain benefits not available from...
See also: computer program

**program construct. (1)** set of one or more procedure parts and a control part which may be implicit (ISO/IEC/IEEE 24765i:2020) Note: Each procedure part consists of one or more operations to be performed or can be null.

**program design language (PDL). (1)** specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a program design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: hardware design language, pseudo code

**Program Evaluation and Review Technique (PERT). (1)** technique for estimating that applies a weighted average of optimistic, pessimistic, and most likely estimates when there is uncertainty with the individual activity estimates (ISO/IEC/IEEE 24765h:2019)

**program instruction. (1)** computer instruction in a source program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A program instruction is distinguished from a computer instruction that results from assembly, compilation, or other interpretation process.

**program librarian. (1)** person responsible for establishing, controlling, and maintaining a software development library (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**program listing. (1)** printout or other human readable display of the source and, sometimes, object statements that make up a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**program maintenance manual. (1)** document that provides the information necessary to maintain a program (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**program management. (1)** the application of knowledge, skills, tools, and techniques to a program to meet the program requirements and to obtain benefits and control not available by managing program components individually (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**program mutation. (1)** computer program that has been purposely altered from the intended version to evaluate the ability of test cases to detect the alteration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

(2) process of creating an altered program as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: mutation testing

**program network chart. (1)** diagram that shows the relationship between two or more computer programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**program specification. (1)** document that describes the structure and functions of a program in sufficient detail to permit programming and to facilitate maintenance (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**program status word (PSW). (1)** computer word that contains information specifying the current status of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) special-purpose register that contains a program status word as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The information can include error indicators, the address of the next instruction to be executed, or currently enabled interrupts.

**program synthesis. (1)** use of software tools to aid in the transformation of a program specification into a program that realizes that specification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
program-sensitive fault. (1) fault that causes a failure when some particular sequence of program steps is executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary). See also: data-sensitive fault

programmable breakpoint. (1) breakpoint that automatically invokes a previously specified debugging process when initiated (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary). See also: code breakpoint, data breakpoint, dynamic breakpoint, epilog breakpoint, prolog breakpoint, static breakpoint

programmable counter array. (1) group of counter modules on a microcontroller unit that increases its timing capabilities (ISO/IEC/IEEE 24765d:2015)

programmable pulse generator. (1) part of a microcontroller unit that produces clock signals, adjustable through programming (ISO/IEC/IEEE 24765d:2015)

programmable read-only memory (PROM). (1) read-only memory unit without a rewrite or an erase data function, which the user can program only once (ISO/IEC/IEEE 24765c:2014) Syn: programmable read only memory

programmable reload timer (PRT). (1) reconfigurable device to restart a function, using decrementing, status control, and reload registers (ISO/IEC/IEEE 24765e:2015)

programmable terminal. (1) user terminal that has built-in data processing capability (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: intelligent terminal

programmatic reference point. (1) reference point at which a programmatic interface can be established to allow access to a function (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 15.3.1)

programmer manual. (1) document that provides the information necessary to develop or modify software for a given computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Typically described are the equipment configuration, operational characteristics, programming features, input/output features, and compilation or assembly features of the computer system. See also: diagnostic manual, installation manual, operator manual, support manual, user manual

programming. (1) general activity of software development (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) designing, writing, modifying, and testing of programs (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: especially construction activities. See also: construction

programming language. (1) language used to express computer programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) artificial language for expressing programs (ISO/IEC 2382:2015 Information technology -- Vocabulary)

programming support environment. (1) integrated collection of software tools accessed via a single command language to provide programming support capabilities throughout the software life cycle (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: sometimes called integrated programming support environment. The environment typically includes tools for specifying, designing, editing, compiling, loading, testing, configuration management, and project management. See also: scaffolding

programming system. (1) set of programming languages and the support software (editors, compilers, linkers, etc.) necessary for using these languages with a given computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
progressive elaboration. (1) the iterative process of increasing the level of detail in a project management plan as greater amounts of information and more accurate estimates become available *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

prohibition. (1) prescription that a particular behavior must not occur *(ISO/IEC 10746-2:2009 Information technology - Open Distributed Processing -- Reference Model: Foundations, 11.2.6)*

project. (1) endeavor with defined start and finish criteria undertaken to create a product or service in accordance with specified resources and requirements *(ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.37) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.33) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.35) (2) undertaking with pre-specified objectives, magnitude and duration *(ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) a temporary endeavor undertaken to create a unique product, service, or result *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (4) set of activities for developing a new product or enhancing an existing product *(ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.38) (5) set of work activities, both technical and managerial, required to satisfy the terms and conditions of a project agreement Note: A project is sometimes viewed as a unique process comprising coordinated and controlled activities and can be composed of activities from the Project Processes and Technical Processes.*

project balance. (1) representation, as a series of cash amounts at regular intervals, of the cumulative to-date value of an alternative *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

project calendar. (1) a calendar that identifies working days and shifts that are available for scheduled activities *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: resource calendar*

project charter. (1) a document issued by the project initiator or sponsor that formally authorizes the existence of a project, and provides the project manager with the authority to apply organizational resources to project activities *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

project communications management. (1) Project communications management includes the processes that are required to ensure timely and appropriate planning, collection, creation, distribution, storage, retrieval, management, control, monitoring, and the ultimate disposition of project information. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: information management*

project control. (1) activities concerned with monitoring the progress of a project, its direction, quality, and resource utilization, as compared with project plan *(ISO/IEC 2382:2015 Information technology -- Vocabulary)*

project cost management. (1) Project cost management includes the processes involved in planning, estimating, budgeting, financing, funding, managing, and controlling costs so that the project can be completed within the approved budget. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

project file. (1) central repository of material pertinent to a project *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Contents typically include memos, plans, technical reports, and related items. Syn: project notebook*

project functional size. (1) the size of a development project or an enhancement project expressed in function
In other words, the total functionality to be added, changed, or deleted (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis, B) Note: It enables those involved to determine the effort required in order to realize new software or to change the functionality of existing software. In the latter case, a project functional size pertains to the addition, change, or deletion of functions.

**project funding requirements.** (1) forecast project costs to be paid for that are derived from the cost baseline for total or periodic requirements, including projected expenditures plus anticipated liabilities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project governance.** (1) the framework, functions, and processes that guide project management activities in order to create a unique product, service, or result to meet organizational, strategic, and operational goals. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project human resource management.** (1) Project Human Resource Management includes the processes that organize, manage, and lead the project team. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project initiation.** (1) launching a process that can result in the authorization of a new project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project integration management.** (1) Project integration management includes the processes and activities needed to identify, define, combine, unify, and coordinate the various processes and project management activities within the project management process groups. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project life cycle.** (1) the series of phases that a project passes through from its start to its completion (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project management (PM).** (1) the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) activities concerned with project planning and project control (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**Project Management Body of Knowledge (PMBOK®).** (1) a term that describes the knowledge within the profession of project management. The project management body of knowledge includes proven traditional practices that are widely applied as well as innovative practices that are emerging in the profession. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project management information system (PMIS).** (1) an information system consisting of the tools and techniques used to gather, integrate, and disseminate the outputs of project management processes (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: It is used to support all aspects of the project from initiating through closing, and can include both manual and automated systems.

**project management knowledge area.** (1) an identified area of project management defined by its knowledge requirements and described in terms of its component processes, practices, inputs, outputs, tools, and techniques (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
project management office (PMO). (1) an organizational structure that standardizes the project-related governance processes and facilitates the sharing of resources, methodologies, tools and techniques. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project management plan. (1) the document that describes how the project will be executed, monitored and controlled, and closed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) information item that describes how the project will be executed, monitored, and controlled (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.11)

project management process group. (1) a logical grouping of project management inputs, tools and techniques, and outputs. The project management process groups include initiating processes, planning processes, executing processes, monitoring and controlling processes, and closing processes. Project management process groups are not project phases. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project management software. (1) applications specifically designed to aid the project management team with planning, monitoring, and controlling the project, including cost estimating, scheduling, collaboration, and risk analysis (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

project management system. (1) the aggregation of the processes, tools, techniques, methodologies, resources, and procedures to manage a project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project management team. (1) the members of the project team who are directly involved in project management activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: project team

project manager (PM). (1) the person assigned by the performing organization to lead the team that is responsible for achieving the project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) person with overall responsibility for the management and running of a project (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.39) (3) stakeholder with overall responsibility for the planning, execution, and closure of a project (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.10) Syn: PJM

project organization chart. (1) a document that graphically depicts the project team members and their interrelationships for a specific project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project performance. (1) derived measure that gives an indication of some attribute associated with how well, how quickly, how effectively, or how efficiently a project is carried out (ISO/IEC 29155-1:2017 Systems and software engineering--Information technology project performance benchmarking framework--Part 1: Concepts and definitions, 2.8)

project phase. (1) a collection of logically related project activities that culminates in the completion of one or more deliverables (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: stage

project plan. (1) document that describes the technical and management approach to be followed for a project (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: for example, a software development plan. The plan typically describes the work to be done, the resources required, the methods to be used, the procedures to
be followed, the schedules to be met, and the way that the project will be organized.

**project planning.** (1) activities concerned with the specification of the components, timing, resources, and procedures of a project (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**project portfolio.** (1) collection of projects that addresses the strategic objectives of the organization (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.38)

**project portfolio management.** (1) centralized management of one or more portfolios of projects to achieve strategic objectives (ISO/IEC 29110-4:2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.1.3) Syn: PPM

**project procurement management.** (1) Project procurement management includes the processes necessary to purchase or acquire products, services, or results needed from outside the project team. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project quality management.** (1) Project Quality Management includes the processes for incorporating the organization?s quality policy regarding planning, managing, and controlling project and product quality requirements, in order to meet stakeholders' expectations. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project resource constraint.** (1) limitation or restraint placed on resource usage, such as what resource skills or disciplines are available and the amount of a given resource available during a specified time frame (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**project resource management.** (1) Project resource management includes the processes to identify, acquire, and manage the resources needed for the successful completion of the project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project risk.** (1) risk related to the management of a project (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.31)

**project risk management.** (1) project risk management includes the processes of conducting risk management planning, identification, analysis, response planning, and monitoring risk on a project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project schedule.** (1) an output of a schedule model that presents linked activities with planned dates, durations, milestones, and resources. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project schedule management.** (1) Project schedule management includes the processes required to manage the timely completion of the project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**project schedule network diagram.** (1) a graphical display of the logical relationships among the project schedule activities. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: activity network diagram

**project scope.** (1) the work that must be performed to deliver a product, service, or result with the specified features and functions. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
project scope management. (1) Project scope management includes the processes required to ensure that the project includes all the work required, and only the work required, to complete the project successfully. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project scope statement. (1) the description of the project scope, major deliverables, assumptions, and constraints (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project specification. (1) specification of the objectives, requirements, and scope of a project and its relations to other projects (ISO/IEC 2382:2015 Information technology -- Vocabulary)

project stakeholder management. (1) Project stakeholder management includes the processes required to identify all people or organizations impacted by the project, analyzing stakeholder expectations and impact on the project, and developing appropriate management strategies for effectively engaging stakeholders in project decisions and execution. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

project team. (1) project manager, and, for some projects, the project sponsor, and the group of persons who report either directly or indirectly to the project manager and who are responsible for performing project work as a regular part of their assigned duties, including the staff supporting project management (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) a set of individuals who support the project manager in performing the work of the project to achieve its objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: project management team

project team directory. (1) a documented list of project team members, their project roles and communication information. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

projectized organization. (1) organizational structure in which the project manager has full authority to assign priorities, apply resources, and direct the work of persons assigned to the project (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

prolog breakpoint. (1) breakpoint that is initiated upon entry into a program or routine (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: preamble breakpoint See also: epilog breakpoint, code breakpoint, data breakpoint, dynamic breakpoint, programmable breakpoint, static breakpoint

PROM. (1) programmable read-only memory (ISO/IEC/IEEE 24765c:2014)

prompt. (1) symbol or message displayed by a computer system, requesting input from the user of the system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to display a symbol or message as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) visual or audible message sent by a program to request or guide the user's response (ISO/IEC 2382:2015 Information technology -- Vocabulary)

proof of concept. (1) realization of an idea or technology to demonstrate its feasibility (INCOSE Systems Engineering Handbook, 4th ed.)

proof of correctness. (1) formal technique used to prove mathematically that a computer program satisfies its specified requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) proof that results from applying the technique in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assertion, formal specification, inductive assertion method, partial correctness, total correctness

property. (1) responsibility that is an inherent or distinctive characteristic or trait that manifests some aspect of an object's knowledge or behavior (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax)
and Semantics for IDEF1X97 (IDEFobject), 3.1.151) Note: Three kinds of property are defined: attribute, participant properties due to relationships, and operations.

**property to quantify.** (1) property of a target entity that is related to a quality measure element and which can be quantified by a measurement method (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.11)

**property-of-interest.** (1) any property that, if lost, is considered a negative effect (ISO/IEC 15026-3:2015 Systems and software engineering -- Systems and software assurance -- Part 3: System integrity levels, 1) (2) object whose loss is considered as a negative effect (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.12) Note: The concept of property-of-interest is introduced in order to characterize negative effects of consequences. In the safety context, human lives and health are instances of property-of-interest. Assets in the security context are instances of properties-of-interest.

**proposal.** (1) supplier's offer to provide a system or service, usually including benefits, costs, risks, opportunities, and other factors applicable to decisions (ISO/IEC/IEEE 24765c:2014) Note: includes business cases

**proposal evaluation techniques.** (1) the process of reviewing proposals provided by suppliers to support contract award decisions (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**proposed change.** (1) report of anomaly, required or recommended enhancement from the time an idea is recorded until the disposition by a designated change authority (ISO/IEC/IEEE 24765i:2020) Note: The disposition can be to reject, to defer for further analysis, or to accept. Upon acceptance the proposed change becomes an approved modification. There can be a one-to-one, one-to-many, or many-to-many relationship between proposed changes and approved modifications.

**proposition.** (1) observable fact or state of affairs involving one or more entities, of which it is possible to assert or deny that it holds for those entities (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 6.2)

**protected.** (1) responsibility that is visible only to the class or the receiving instance of the class (available only within methods of the class or its subclasses) (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.152) See also: private, public, hidden

**protection.** (1) process to secure content (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.12)

**protection exception.** (1) exception that occurs when a program attempts to write into a protected area in storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: addressing exception, data exception, operation exception, overflow exception, underflow exception

**protocol.** (1) set of conventions that govern the interaction of processes, devices, and other components within a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**protocol object.** (1) engineering object in a channel, which communicates with other protocol objects in the same channel to achieve interaction between basic engineering objects (possibly in different clusters, possibly in different capsules, possibly in different nodes) (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.12)
prototype. (1) preliminary type, form, or instance of a system that serves as a model for later stages or for the final, complete version of the system (ISO/IEC/IEEE 24765a:2011) (2) an experimental model, either functional or nonfunctional, of the system or part of the system (3) a method of obtaining early feedback on requirements by providing a working model of the expected product before actually building it (A Guide to the Project Management Body of Knowledge (PMBOK (R) Guide) -- Sixth Edition) (4) production#&8208;ready demonstration model developed under engineering supervision that is specification compliant and represents what manufacturing should replicate (INCOSE Systems Engineering Handbook, 4th ed.) Note: A prototype is used to get feedback from users for improving and specifying a complex human interface, for feasibility studies, or for identifying requirements.

prototyping. (1) hardware and software development technique in which a preliminary version of part or all of the hardware or software is developed to permit user feedback, determine feasibility, or investigate timing or other issues in support of the development process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: rapid prototyping

provision. (1) expression in the content of a normative document, that takes the form of a statement, an instruction, a recommendation or a requirement (ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.8) Note: These types of provision are distinguished [in English] by the form of wording they employ, e.g., instructions are expressed in the imperative mood, recommendations by the use of the auxiliary "should", and requirements by the use of the auxiliary "shall". [ISO/IEC Guide 2:2004]

proxy home. (1) implementation of the component home interface specified by a composition definition (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: The implementation is not required to be collocated with the container where the components managed by the home are activated.

PRR. (1) production readiness review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

PRT. (1) programmable reload timer (ISO/IEC/IEEE 24765e:2015)


pseudo instruction. (1) source language instruction that provides information or direction to the assembler or compiler and is not translated into a target language instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: pragma, pseudo-op, pseudo operation

pseudo-oracle. (1) 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 (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.59) Syn: derived test oracle

pseudocode. (1) combination of programming language constructs and natural language used to express a computer program design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) general term for

**pseudostatic random access memory (PSRAM).** (1) dynamic random access memory unit with an automatic refresh circuit on the unit (ISO/IEC/IEEE 24765c:2014)


**PSP.** (1) product support plan (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**PSRAM.** (1) pseudostatic random access memory (ISO/IEC/IEEE 24765c:2014)

**PSW.** (1) program status word (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**psychometrics.** (1) field of study concerned with the theory and technique for developing valid and reliable psychological measures (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.14)

**PT.** (1) practitioner (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.21)

**public.** (1) responsibility that is not hidden (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.153) (2) known to multiple routines or modules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: That is, visible to any requester (available to all without restriction). See also: hidden, private, protected

**publisher.** (1) event source that can be connected to an arbitrary number of event sinks, which subscribe to the publisher event source (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

**publishing pipeline.** (1) series of defined processing steps that are connected to transform content from its source format into a final deliverable format (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.27)

**pulse width modulation (PWM).** (1) control of electrical signals with various cycle and duty ratios (ISO/IEC/IEEE 24765d:2015) Syn: pulse-width modulation

**purpose of the count.** (1) reason for performing the function point count (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.45)

**purpose statement.** (1) brief statement of the reason for an IDEF0 model's existence that is presented in the a-0 context diagram of the model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.96)

**PV.** (1) planned value (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

QA. (1) quality assurance (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)

QC. (1) quality control (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


QE. (1) quality engineering (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.3)


QFP. (1) quad flat pack (ISO/IEC/IEEE 24765c:2014)


QM-RM. (1) quality measurement reference model (ISO/IEC 25020:2019 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Quality measurement framework, 4)

QME. (1) quality measure element (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 5)


QOS. (1) Quality of Service (ISO/IEC 19793:2015 Information technology -- Open Distributed Processing -- Use of UML for ODP system specifications, 4)

QR. (1) quality requirements (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 4)

QTFF. (1) QuickTime File Format (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 5) Note: The file extension is .mov

qualification. (1) process of demonstrating whether an entity is capable of fulfilling specified requirements (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.39) (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.36) (2) process of determining whether a system or component is suitable for operational use (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) generally, demonstrated education, training and work experience, where applicable (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.13) (4) specific professional recognition, title or token, which may indicate proficiency, skill or knowledge in a given domain, but which is done on a one-time basis only (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.14) Note: The process can include testing and analyzing hardware and software configuration items to prove that the design will survive the anticipated accumulation of acceptance test environments, plus its expected handling, storage, and operational environments, plus a specified qualification margin.


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
qualification body. (1) entity issuing certificates of qualification (ISO/IEC/IEEE 24765h:2019)

qualification scheme. (1) requirements which, when satisfied, result in the issuance of a qualification (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.15)

qualification testing. (1) testing conducted to determine whether a system or component is suitable for operational use (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) testing conducted on a hardware element, software element, or system to evaluate conformance with specified requirements (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) See also: acceptance testing, development testing, operational testing

qualitative risk analysis. (1) prioritizing risks for subsequent further analysis or action by assessing and combining their probability of occurrence and impact (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

quality. (1) degree to which the system satisfies the stated and implied needs of its various stakeholders, and thus provides value (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 3.1) (2) ability of a product, service, system, component, or process to meet customer or user needs, expectations, or requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) the degree to which a set of inherent characteristics fulfils requirements (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

quality assurance (QA). (1) part of quality management focused on providing confidence that quality requirements will be fulfilled (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.40) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.34) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.37) Note: There are both internal and external purposes for quality assurance: within an organization, quality assurance provides confidence to management; in contractual situations, quality assurance provides confidence to the customer or others. Some quality control and quality assurance actions are interrelated. Unless requirements for quality fully reflect the needs of the user, quality assurance does not necessarily provide adequate confidence.

quality attribute. (1) characteristic of software, or a generic term applying to quality factors (ISO/IEC/IEEE 24765:2021)

quality audit. (1) a quality audit is a structured, independent process to determine if project activities comply with organizational and project policies, processes, and procedures. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


quality checklist. (1) a structured tool used to verify that a set of required steps has been performed (A Guide to the
quality control (QC). (1) set of activities designed to evaluate the quality of developed or manufactured products (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) monitoring service performance or product quality, recording results, and recommending necessary changes (ISO/IEC/IEEE 24765c:2014) See also: control quality, quality assurance

quality control measurements. (1) the documented results of control quality activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

quality evaluation. (1) systematic examination of the extent to which an entity is capable of fulfilling specified requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The requirements can be formally specified, as when a product is developed for a specific user under a contract, or specified by the development organization, as when a product is developed for unspecified users, such as consumer software, or the requirements can be more general, as when a user evaluates products for comparison and selection purposes.

quality in use (measure). (1) extent to which a product used by specific users meets their needs to achieve specific goals with effectiveness, productivity, safety and satisfaction in specific contexts of use (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.42) (2) degree to which a product or system can be used by specific users to meet their needs to achieve specific goals with effectiveness, efficiency, freedom from risk and satisfaction in specific contexts of use (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.24) (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.8) (3) extent to which the behavioral and attitudinal outcomes and consequences of use of a product, system, or service meet the needs of users or other stakeholders in specific contexts of use (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.13) Note: This definition of quality in use is similar to the definition of usability in ISO 9241-11. Before the product is released, quality in use can be specified and measured in a test environment designed and used exclusively by the intended users for their goals and contexts of use, e.g. User Acceptance Testing Environment. See also: usability

quality in use measure. (1) measure of the degree to which a product or system can be used by specific users to meet their needs to achieve specific goals with effectiveness, efficiency, freedom from risk, satisfaction and content coverage in specific contexts of use (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.25) (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.12)


quality management plan. (1) a component of the project or program management plan that describes how applicable policies, procedures, and guidelines will be implemented to achieve the quality objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition, 2017). Copyright and all rights reserved. PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
quality management system. (1) The organizational framework whose structure provides the policies, processes, procedures, and resources required to implement the quality management plan. The typical project quality management plan should be compatible to the organization's quality management system. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

quality measure. (1) measure that is defined as a measurement function of two or more values of quality measure elements (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.10) (2) derived measure that is defined as a measurement function of two or more values of quality measure elements (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.13) Syn: QM See also: software quality measure

quality measure element (QME). (1) measure defined in terms of a property and the measurement method for quantifying it, including optionally the transformation by a mathematical function (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.26) (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.14) (2) measure defined in terms of an attribute and the measurement method for quantifying it, including optionally the transformation by a mathematical function (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.11) Note: The software quality characteristics or subcharacteristics of the entity are derived afterwards by calculating a software quality measure.

quality measure on external property. (1) measure of the degree to which a system or software product enables its behavior to satisfy stated and implied needs for the system including the software to be used under specified conditions (ISO/IEC 25020:2019 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Quality measurement framework, 3.15) Syn: QM on external property

quality measure on internal property. (1) measure of the degree to which a set of static attributes of a software product satisfies stated and implied needs for the software product to be used under specified conditions (ISO/IEC 25020:2019 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Quality measurement framework, 3.16) Note: Quality measures on internal property are typically associated with quality requirements on static properties and attributes that can be specified in or derived from requirements. Syn: QM on internal property

quality metric. (1) quantitative measure of the degree to which an item possesses a given quality attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given quality attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

quality metrics. (1) a description of a project or product attribute and how to measure it (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: quality measure

quality model. (1) defined set of characteristics and of relationships between them, which provides a framework for

quality of service. (1) set of quality requirements on the collective behavior of one or more objects (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.2)

quality policy. (1) a policy specific to the Project Quality Management Knowledge Area, it establishes the basic principles that should govern the organization's actions as it implements its system for quality management. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

quality property. (1) measurable component of quality (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.9)

quality record report. (1) report which is generated through dynamic test execution and code analysis to record test results and other output (ISO/IEC 30130:2016 Software engineering --Capabilities of software testing tools, 3.1) Note: including Test Result, Static Code Analysis Report, Test Incident Report, and Metrics.

quality requirement. (1) requirement for quality properties or attributes of an ICT product, data, or service that satisfy needs which ensue from the purpose for which that ICT product, data, or service is to be used (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.15) (2) capability of a product to satisfy the stated and implied needs when used under specific conditions (ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.5) See also: non-functional requirement

quantitative risk analysis. (1) numerical analysis of the effect on overall project objectives of identified risks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

query language. (1) language used to access information stored in a database (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: programming language, specification language

questionnaire. (1) written set of questions designed to quickly accumulate information from a large number of respondents (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: survey

queue. (1) list in which items are appended to the last position of the list and retrieved from the first position of the list (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

quiescing. (1) process of bringing a device or system to a halt by rejecting new requests for work (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


RACI. (1) responsible, accountable, consult, and inform (ISO/IEC/IEEE 24765c:2014) (2) a common type of responsibility assignment matrix that uses responsible, accountable, consult, and inform statuses to define the involvement of stakeholders in project activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: RACI chart

RAM. (1) responsibility assignment matrix (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide)
random-access memory (RAM). (1) volatile semiconductor storage device which allows data to be written or accessed in approximately the same amount of time, regardless of the data's physical location (ISO/IEC/IEEE 24765c:2014) Note: often used for caches or main memory in a computer and embedded in an MCU chip. Compare to a CD where data is stored and accessed sequentially. Syn: random access memory

rapid prototyping. (1) type of prototyping in which emphasis is placed on developing prototypes early in the development process to permit early feedback and analysis in support of the development process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: waterfall model, data structure-centered design, incremental development, input-process-output, modular decomposition

rate-monotonic algorithm. (1) real-time scheduling algorithm that assigns higher priorities to tasks with shorter periods (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

rating. (1) action of mapping the measured value to the appropriate rating level (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.28) Note: used to determine the rating level associated with the software for a specific quality characteristic. Rating and rating levels can be applied to characteristics other than quality characteristics.

rating interval. (1) time interval of the measurement procedure from the time the SUT reaches a stable state of operation to the time the measurement results are fulfilling the required statistical significance (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.13)

rating level. (1) scale point on an ordinal scale, which is used to categorize a measurement scale (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.29) Note: The rating level enables software product to be classified (rated) in accordance with the stated or implied needs. Appropriate rating levels can be associated with the different views of quality, i.e., 'Users', 'Managers', or 'Developers.'

ratio scale. (1) scale in which the measurement values have equal distances corresponding to equal quantities of the attribute where the value of zero corresponds to none of the attribute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: For example, the size of a software component in terms of LOC is a ratio scale, because the value of zero corresponds to no lines of code and each additional increment represents equal amounts of code. See also: interval scale, nominal scale, ordinal scale


RDBMS. (1) Relational Database Management System (ISO/IEC 25024:2015 Systems and software engineering -
Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 5)

RDF. (1) Resource Description Framework (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)


RE. (1) service relationship process (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecyle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 4.1)

reachability graph. (1) directed graph of nodes and edges, where the nodes correspond to reachable markings, and the edges correspond to transition occurrences (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.30)

reachable marking. (1) marking of the net that can be reached from the initial marking by the occurrence of transitions (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.31)

reactivation. (1) cloning a cluster following its deactivation (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.26)

reactive approach. (1) approach of developing a product line or product variations in response to stated needs or customer requirements (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.16)

reactiveness. (1) degree to which the IT service promptly responds to user requests (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.6.2)

read. (1) to access data from a storage device or data medium (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) data movement type that moves a data group from persistent storage within reach of the functional process which requires it (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.23) Note: A read is considered to include certain associated data manipulations necessary to achieve the read. Syn: read type See also: destructive read, nondestructive read, write

read-only. (1) property that causes no state changes (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.155) Note: That is, it does no updates. Syn: read only

read-only memory (ROM). (1) non-volatile semiconductor storage device, from which data cannot be removed once it is written (ISO/IEC/IEEE 24765d:2015) Syn: read only memory

readability. (1) ease with which a system's source code can be read and understood, especially at the detailed, statement level (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

reader note. (1) comment made by a reader about a diagram and placed on the diagram page (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.97) Note: A reader note is not part of the diagram itself, but rather is used for communication about a diagram during model development.

reading reference. (1) data storage entity or record, or interface record from another software or system containing...
data retrieved in a BFC (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FiSMA 1.1 functional size measurement method, 3.8) Note: The number of reading references is equal to 0 for all BFC types where it is applicable.

ready to use software product (RUSP). (1) software product available to any user, at cost or not, to use without the need to conduct development activities (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.6) Note: includes the product description (including cover information, data sheet, and website information; the user documentation necessary to install and use the software, including any configuration of the operating system or target computer required to operate the product; the software contained on a computer sensible media (e.g., disk or CD-ROM) or internet downloadable. Includes software produced and supported without typical commercial fees and licensing considerations. Syn: ready-to-use software product See also: commercial off the shelf (COTS)

real address. (1) address of a storage location in the main storage part of a virtual storage system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: virtual address

real storage. (1) main storage portion of a virtual storage system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: virtual storage

real type. (1) data type whose members can assume real numbers as values and can be operated on by real number arithmetic operations, such as addition, subtraction, multiplication, division, and square root (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: character type, enumeration type, integer type, logical type

real-time. (1) problem, system, or application that is concurrent and has timing constraints whereby incoming events must be processed within a given timeframe (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: realtime, real time


real-time operating system (RTOS). (1) operating system intended to handle transaction requests immediately upon receipt (ISO/IEC/IEEE 24765c:2014)

real-time scheduling theory. (1) theory for priority-based scheduling of concurrent tasks with hard deadlines (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It addresses how to determine whether a group of tasks, whose individual CPU utilization is known, will meet their deadlines.

real-world object. (1) entity that exists in a three-dimensional form and, by association, implies similar properties or behavior to software functions (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.41)

realization states "how" a responsibility is met; it is the statement of the responsibility's method. Realization consists of any necessary representation properties together with the algorithm (if any). A realization can involve representation properties or an algorithm, or both.

reasonably foreseeable misuse. (1) use of a product or system in a way not intended by the supplier, but which can result from readily predictable human behavior (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.30) Note: Readily predictable human behavior includes the behavior of all types of users.

reasoning technique. (1) form of artificial intelligence that generates conclusions from available information using logical techniques, such as deduction and induction (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.60)

recall. (1) performance metric used to evaluate a classifier that measures the proportion of actual positives that were predicted correctly (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.61) Syn: sensitivity

receptacle. (1) operation interface in which a computational component plays a client role (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.16) (2) named connection point that describes the component's ability to use a reference supplied by some external agent (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)


record element type (RET). (1) user-recognizable sub-group of data element types within a data function (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.46) (2) user recognizable sub-group of data element types (DETs) within a data function, and code data group (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1)

record type. (1) an entity type in a logical file (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Syn: RET

records management system. (1) processes, related control functions, and tools that identify, store, retrieve, and retain data (ISO/IEC/IEEE 24765h:2019)

recoverability. (1) degree to which, in the event of an interruption or a failure, a product or system can recover the
data directly affected and re-establish the desired state of the system (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.5.4) (2) degree to which object state changes resulting from failed transactions are cancelled (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.7.1.4) See also: survivability

recovery. (1) restoration of a system, program, database, or other system resource to a state in which it can perform required functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) cloning a cluster after cluster failure or deletion (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.25) See also: backward recovery, checkpoint, forward recovery

recursion. (1) process in which a software module calls itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of defining or generating a process or data structure in terms of itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: simultaneous recursion

recursive. (1) pertaining to a software module that calls itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to a process or data structure that is defined or generated in terms of itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: reflexive

redundancy. (1) in fault tolerance, the presence of auxiliary components in a system to perform the same or similar functions as other elements for the purpose of preventing or recovering from failures (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: active redundancy, diversity, homogeneous redundancy, standby redundancy

reengineering. (1) examination and alteration of software to reconstitute it in a new form, including the subsequent implementation of the new form (ISO/IEC TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK), 5.4.2) (2) complete cycle of performing reverse engineering followed by forward engineering (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

reentrant. (1) pertaining to a software module that can be entered as part of one process while also in execution as part of another process and still achieve the desired results (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: reenterable, re-entrant

reentry point. (1) place in a software module at which the module is reentered following a call to another module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: re-entry point

refactor. (1) to restructure software code without altering its behavior for the purpose of improving quality attributes, easing future extension or adaptation, or adhering to an architectural style (Software Extension to the PMBOK(R) Guide Fifth Edition)

reference architecture. (1) core architecture that captures the high-level architecture concept of domain architecture and application architecture (ISO/IEC 26552:2019 Software and systems engineering--Tools and methods for product line architecture design, 3.9)

reference body of knowledge. (1) body of knowledge that is used for the comparison of a particular body of knowledge associated with a certification scheme (ISO/IEC 24773-1:2019 Software and systems engineering-Certification of software and systems engineering professionals-Part 1: General requirements, 3.12)


This definition is copyrighted ©, 2021 by the IEEE.
The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition.
All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
reference expression. (1) expression that uniquely identifies a box, a node or function, a diagram, or a model page within an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.98)

reference FSM method. (1) an FSM method to be used for comparison reasons when verifying the functional size measurement results (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 3.3)

reference mode. (1) usage mode that is intended to provide quick access to specific information for software users who are generally familiar with the software functions (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.42)

reference node. (1) node of a Petri net that is a representative of another node, possibly defined on another page of the net graph (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.14)

reference place. (1) reference node that represents a place and refers to either another reference place or to a place (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.15)

reference point. (1) interaction point defined in an architecture for selection as a conformance point in a specification which is compliant with that architecture (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 10.6)

reference transition. (1) reference node that represents a transition and refers to either another reference transition to a transition (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.16)

reference user requirement collection (RUR Collection). (1) a subset of RUR which is selected to match the purpose in a specific evaluation (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 3.5)

reference user requirements (RUR). (1) a standard set of user requirements which conforms to the requirements (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 3.4)


refinement. (1) process of transforming one specification into a more detailed specification (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.6)

reflective construct. (1) construct that is viewed as the cause of measures in the relationship between a construct and its measures (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.13)

reflexive. (1) in a relationship, the condition when the same data object plays both (binary or many (n-ary)) roles (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.5) See also: recursive
reflexive ancestor (of a class). (1) class itself or any of its generic ancestors *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.158) See also: generic ancestor

refresh. (1) method to keep data in volatile memory by rewriting the data before it disappears from the memory *(ISO/IEC/IEEE 24765c:2014) Note: needed for memory with a circuit architecture in single stable state

Regex. (1) regular expression *(ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.28)

regid. (1) registration identifier *(ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.27)

register. (1) small storage unit in a processing unit *(ISO/IEC/IEEE 24765c:2014) Note: Registers can be set up in CPU, microcontroller, digital signal processor, or microprocessor.


registration identifier. (1) unique identifier for an entity *(ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.27) Syn: regid

registry. (1) book or system for keeping an official list or record of work products and the associated information items *(ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.18) Note: Repository and library items can be recorded in registries to enable better management and governance of these items.

regression. (1) machine learning function that results in a numerical or continuous output value for a given input *(ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.62)

regression analysis. (1) an analytic technique where a series of input variables are examined in relation to their corresponding output results in order to develop a mathematical or statistical relationship. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

regression test. (1) retesting to detect faults introduced by modification *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

regression testing. (1) 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 *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) testing following modifications to a test item or to its operational environment, to identify whether regression failures occur *(ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.32)

regular expression (Regex). (1) string of characters that allows patterns to be used to match search results *(ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.28) Note: Patterns may dictate that matches must start or end with specific sequences of characters or allow the use of wildcards to match any characters in a sequence.

regulation. (1) requirements imposed by a governmental body. These requirements can establish product, process or service characteristics, including applicable administrative provisions that have government-mandated compliance. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

reinforcement learning. (1) task of building a ML model using a process of trial and reward to achieve an objective (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.64) Note: A reinforcement learning task can include the training of a machine learning model in a way similar to supervised learning, plus training on unlabeled inputs gathered during the operation phase of the AI system. Each time the model makes a prediction, a reward is calculated, and further trials are run to optimize the reward. In reinforcement learning, the objective, or definition of success, can be defined by the system designer. In reinforcement learning, the reward can be a calculated number that represents how close the artificial intelligence system is to achieving the objective for a given trial.

relation. (1) set of relationships of the same relationship type (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 3.2.3) (2) association between two or more domains of entities (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.13)

relational database management system. (1) management system for relational database (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.34) Note: In order to use relational data base management systems (RDBMS), it is necessary to represent relational model of data that organizes data with specific characteristics (tables or relations, unique key, etc.)

relationship. (1) real-world association among one or more entities (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (2) association between two (not necessarily distinct) classes that is deemed relevant within a particular scope and purpose (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.159) (3) semantic connection between model elements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) association between two or more entities (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.14) (5) predicate involving two or more roles with assigned values (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 3.2.1) Note: The association is named for the sense in which the instances are related. A relationship can be represented as a time-varying binary relation between the instances of the current extents of two state classes.


relationship manager (RM). (1) role that develops and manages the customer and supplier interfaces as well as the service catalogue (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.26) Note: This role may be combined with other roles. This role is a direct report (or shared role) with the Service Manager.

relationship name. (1) verb or verb phrase that reflects the meaning of the relationship expressed between the two entities shown on the diagram on which the name appears (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.161) Note: [key style]

relationship type. (1) type of relationship which expresses the number and type of the roles (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 3.2.1)
relative address. (1) address that must be adjusted by the addition of an offset to determine the address of the storage location to be accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: absolute address, base address, indexed address, self-relative address.

relative chain frequency. (1) relative frequency of using the l-th chain type by an emulated user of the i-th type (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.14)


release. (1) particular version of a configuration item that is made available for a specific purpose (ISO/IEC/IEEE 90003:2018 Software engineering -- Guidelines for the application of ISO 9001:2015 to computer software, 3.10) (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.12) (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.43) (2) collection of new or changed configuration items that are tested and introduced into a live environment together (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (3) collection of one or more new or changed configuration items deployed into the live environment as a result of one or more changes (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.28) (4) software version that is made formally available to a wider community (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (5) delivered version of an application which includes all or part of an application (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (6) set of grouped change requests, established in the Application Change Management Process, which are designed, developed, tested, and deployed as a cohesive whole (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.28) Note: Release management includes defining acceptable quality levels for release, authority to authorize the release, and release procedures. See also: version

release engineer. (1) person responsible for coordinating development toward a release (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: The release engineer will monitor pending issues for a given release, oversee the code freeze, and tag the release once it gets out the door.

release map. (1) a displayed forecast of when software features will be released and how they will be grouped into releases (Software Extension to the PMBOK(R) Guide Fifth Edition)

release plan. (1) plan that describes what portions of system functionality will be implemented in which releases and the rationale for each release (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) Note: It includes or provides reference to a description of release contents, release schedule, release impacts, and release notifications.

relevant stakeholder. (1) stakeholder that is identified for involvement in specified activities and is included in a plan (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

reliability. (1) ability of a system or component to perform its required functions under stated conditions for a specified period of time (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) degree to which a system,
product or component performs specified functions under specified conditions for a specified period of time (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.5) (3) degree to which an object or an object’s services provide agreed or expected functionality during a defined time period under specified conditions (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.29) Note: Dependability characteristics include availability and its inherent or external influencing factors, such as availability, reliability (including fault tolerance and recoverability), security (including confidentiality and integrity), maintainability, durability, and maintenance support. Wear or aging does not occur in software. Limitations in reliability are due to faults in requirements, design, and implementation, or due to contextual changes. See also: availability, MTBF

reliability growth. (1) improvement in reliability that results from correction of faults (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

reliability testing. (1) type of testing conducted to evaluate the ability of a test item to perform its required functions, including evaluating the frequency with which failures occur, when it is used under stated conditions for a specified period of time (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.33)

relocatable. (1) pertaining to code that can be loaded into any part of main memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The starting address is established by the loader, which then adjusts the addresses in the code to reflect the storage locations into which the code has been loaded. See also: relocating loader

relocatable address. (1) address that is to be adjusted by the loader when the computer program containing the address is loaded into memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: absolute address

relocatable code. (1) code containing addresses that are to be adjusted by the loader to reflect the storage locations into which the code is loaded (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: absolute code

relocate. (1) to move machine code from one portion of main memory to another and to adjust the addresses so that the code can be executed in its new location (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

relocating assembler. (1) assembler that produces relocatable code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: absolute assembler

relocating loader. (1) loader that reads relocatable code into main memory and adjusts the addresses in the code to reflect the storage locations into which the code has been loaded (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: relative loader See also: absolute loader

relocation dictionary. (1) part of an object module or load module that identifies the addresses that must be adjusted when a relocation occurs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

relocation transparency. (1) distribution transparency which masks relocation of an interface from other interfaces bound to it (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.4.1.5)
relocator. (1) object that manages a repository of locations for interfaces, including locations of management functions for the cluster supporting those interfaces (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 14.3.1.1)

remote job entry (RJE). (1) submission of jobs through a remote input device connected to a computer through a data link (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: remote batch entry remote terminal emulator (RTE). (1) data processing system realizing a set of emulated users (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.15)

repair. (1) correction of defects that have resulted from errors in external design, internal design, or code (ISO/IEC/IEEE 24765a:2011) (2) corrective maintenance of defective or damaged parts or functions of a product (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.31) Syn: defect removal

reparent. (1) to move the changes developed under one branch into another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The changes are not committed to the original branch.

repeatability (of results of measurements). (1) closeness of the agreement between the results of successive measurements of the same measurand carried out under the same conditions of measurement (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.8) (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.15) Note: These conditions are called repeatability conditions. Repeatability conditions include the same measurement procedure, the same observer, the same measuring instrument, used under the same conditions; the same location; repetition over a short period of time. Repeatability is expressed quantitatively in terms of the dispersion characteristics of the results.

repetitive addressing. (1) method of implied addressing in which the operation field of a computer instruction is understood to address the operands of the last instruction executed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-ahead addressing

replaceability. (1) degree to which a product can replace another specified software product for the same purpose in the same environment (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.8.3) Note: Replaceability of a new version of a software product is important to the user when upgrading. Replaceability will reduce lock-in risk, so that other software products can be used in place of the present one, See also: adaptability, installability

replication. (1) copying a software product from one medium to another (ISO/IEC/IEEE 90003:2018 Software engineering -- Guidelines for the application of ISO 9001:2015 to computer software, 3.13)

replication schema. (1) specification of constraints on the replication of an object including both constraints on the availability of the object and constraints on the performance of the object (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 16.7.1.1)

replication transparency. (1) distribution transparency which masks the use of a group of mutually behaviorally compatible objects to support an interface (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing
replication. (1) changes recorded in the repository that do not contribute anything useful to the project's history.

repo bloat. (1) changes recorded in the repository that do not contribute anything useful to the project's history (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A repo master can reduce repo bloat through repo surgery.

repo surgery. (1) changes made directly to the version control system's repository, bypassing the system's commands (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Through repo surgery, a repo master can perform operations that the version control system does not directly support.

report. (1) information item that describes the results of activities such as investigations, observations, assessments, or tests (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.21) (2) an output of data in a layout specified by the user (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: The output medium used is not relevant for FPA and it can pertain to both an external output and an external inquiry.

report standard. (1) standard that describes the characteristics of describing results of engineering and management activities (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

reporting system. (1) facilities, processes and procedures used to generate or consolidate reports from one or more information management systems and distribute reports to stakeholders (ISO/IEC/IEEE 24765h:2019)

repository. (1) organized and persistent data storage that allows data retrieval (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.24) (2) location/format in which such a collection is stored (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (3) collection of all software-related artifacts belonging to a system (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (4) collection of all system element or software-related artifacts belonging to a system (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.49) (5) place where work products and the associated information items are or can be stored for preservation and retrieval (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.19) Note: Artifacts are, for example, the software engineering environment. Benchmarking repository is a repository which is designated for use as the source of comparative measures for the purpose of benchmarking. In a repository, work products and other items are preserved for future retrieval when needed, whereas in a library, working data is temporarily stored and retrieved as necessary.

repository owner. (1) person or organization that owns and maintains a benchmarking repository (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 3.2) Syn: repository administrator

representation. (1) logical portrayal of a physical, operational, or conceptual image or situation (ISO/IEC/IEEE 24765e:2015) (2) one or more properties used by an algorithm for the realization of a responsibility (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.162)

representation property. (1) property on which an algorithm operates (IEEE 1320.2-1998 (R2004) IEEE Standard
for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.163)

representation standard. (1) standard that describes the characteristics of portraying aspects of an engineering or management product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

reproducibility (of results of measurements). (1) closeness of the agreement between the results of measurements of the same measurand carried out under changed conditions of measurement (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.9) (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.16) Note: A valid statement of reproducibility requires specification of the conditions changed. The changed conditions include the principle of measurement; method of measurement; observer; measuring instrument; reference standard; location; conditions of use; time. Reproducibility can be expressed quantitatively in terms of the dispersion characteristics of the results. Results are here usually understood to be corrected results.

request. (1) a message sent from one object (the sender) to another object (the receiver), directing the receiver to fulfill one of its responsibilities (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.164) (2) message issued by a client to cause a service to be performed (ISO/IEC 19500-2:2012 Information technology--Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.12) (3) information item that initiates a defined course of action or change to fulfill a need (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.22) Note: Specifically, a request can be for the value of an attribute, for the value of a participant property, for the application of an operation, or for the truth of a constraint. Request also encompasses sentences of such requests. Logical sentences about the property values and constraints of objects are used for queries, pre-conditions, post-conditions, and responsibility realizations. See also: message

request for change. (1) proposal for a change to be made to a system, service, component, or the service management system (ISO/IEC/IEEE 24765c:2014) (2) proposal for a functional or non-functional change to be made to an existing application (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.30) Note: A change to a service includes the provision of a new service or the removal of a service which is no longer required. See also: change request, modification request.

request for information (RFI). (1) a type of procurement document whereby the buyer requests a potential seller to provide various pieces of information related to a product or service or seller capability (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

request for proposal (RFP). (1) document used by the acquirer as the means to announce its intention to potential bidders to acquire a specified system, software product, or software service (2) collection of formal documents that includes a description of the desired form of response from a potential supplier, the relevant statement of work for the supplier, and required provisions in the supplier agreement (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) a type of procurement document used to request proposals from prospective sellers of products or services. In some application areas, it may have a narrower or more specific meaning (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: request for tender, solicitation package
request for quotation (RFQ). (1) a type of procurement document used to request price quotations from prospective sellers of common or standard products or services. Sometimes used in place of request for proposal and in some application areas, it may have a narrower or more specific meaning. *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

request form. (1) description or pattern that can be evaluated or performed multiple times to cause the issuing of requests *(ISO/IEC 19500-1:2012 Information technology-- Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3.2)*

required inputs. (1) set of items necessary to perform the minimum verification and validation (V&V) tasks mandated within any life cycle activity *(IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)*

required outputs. (1) set of items produced as a result of performing the minimum verification and validation tasks mandated within any life cycle activity *(IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1)*

requirement. (1) statement that translates or expresses a need and its associated constraints and conditions *(ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 4.1.31) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 3.1.44) (2) condition or capability that must be met or possessed by a system, system component, product, or service to satisfy an agreement, standard, specification, or other formally imposed documents *(IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) (3) provision that conveys criteria to be fulfilled *(ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.10) (4) a condition or capability that is necessary to be present in a product, service, or result to satisfy a business need *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (5) need or expectation that is stated, generally implied, or obligatory *(ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.47)* Note: Requirements exist at different tiers and express the need in high-level form. A requirement is denoted by the word "shall" and includes both the exclusive and applicable optional requirements. Requirements provide value when delivered, satisfied, or met. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders. Well-formed requirements are unambiguous, clear, unique, consistent, stand-alone (not grouped), verifiable, and necessary for stakeholder acceptability. See also: design requirement, functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement

requirement standard. (1) a standard that describes the characteristics of a requirements specification *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

requirements allocation. (1) assignment or budgeting of top-level functional or nonfunctional requirements among the lower-level partitioned functions for accomplishment *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: In this manner, the system elements that perform all or part of specific requirements are identified.

requirements analysis. (1) process of studying user needs to arrive at a definition of system, hardware, or software requirements *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of studying and

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC. In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
refining system, hardware, or software requirements *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)* *(3)* systematic investigation of user requirements to arrive at a definition of a system *(ISO/IEC 2382:2015 Information technology -- Vocabulary)* *(4)* determination of product- or service-specific performance and functional characteristics based on analyses of customer needs, expectations, and constraints; operational concept; projected utilization environments for people, products, services, and processes; and measures of effectiveness *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**requirements attributes.** *(1)* set of properties associated with requirements *(ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.7)*

**requirements derivation.** *(1)* changing or translation of a requirement through analysis into a form that is suitable for low-level analysis or design *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* in a hierarchical structure, the next lower level that is associated with a given element *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**requirements document.** *(1)* document containing any combination of requirements or regulations to be met by a ready to use software product (RUSP) *(ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuARE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.13)* See also: requirements documentation

**requirements documentation.** *(1)* a description of how individual requirements meet the business need for the project *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**requirements elicitation.** *(1)* use of systematic techniques, such as prototyping and structured surveys, to proactively identify and document customer and end-user needs *(ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.1.20)*

**requirements engineering.** *(1)* interdisciplinary function that mediates between the domains of the acquirer and supplier to establish and maintain the requirements to be met by the system, software or service of interest *(ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.19)* Note: Requirements engineering is concerned with discovering, eliciting, developing, analyzing, determining verification methods, validating, communicating, documenting, and managing requirements See also: software requirements engineering

**requirements flow-down.** *(1)* systematic decomposition of system requirements into allocated and derived requirements, appropriately assigned to low-level functional components *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**requirements management.** *(1)* activities that ensure requirements are identified, documented, maintained, communicated and traced throughout the life cycle of a system, product, or service *(ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.20)* *(2)* provision of storing and editing capabilities, tracking history of edition, versioning, author identification, change management, time stamping, user notification for content changes, security rights control *(ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.3)* See also: software requirements management
requirements management plan. (1) a component of the project or program management plan that describes how requirements will be analyzed, documented, and managed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

requirements partitioning. (1) separation or decomposing of a top-level requirement or design into successively lower-level detailed requirements or design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

See also: decomposition

requirements phase. (1) period of time in the software life cycle during which the requirements for a software product are defined and documented (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

requirements review. (1) process or meeting during which the requirements for a system, hardware item, or software item are presented to project personnel, managers, users, customers, or other interested parties for comment or approval (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include system requirements review, software requirements review. See also: code review, design review, formal qualification review, test readiness review

requirements specification. (1) document that specifies the requirements for a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards. See also: design description, functional specification, performance specification

requirements specification language. (1) specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document hardware or software requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: design language

requirements traceability. (1) identification and documentation of the derivation path (upward) and allocation/ flow-down path (downward) of requirements in the requirements set (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.1.23) (2) discernible association between a requirement and related requirements, implementations, and verifications (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) traceabilities in domain and application requirements respectively and those between them (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.18)

requirements traceability matrix (RTM). (1) structured information artifact that links requirements to their higher-level requirements or needs or to lower-level implementation (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.1.24) (2) a grid that links product requirements from their origin to the deliverables that satisfy them (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

requirements traceability tool. (1) software development tool that establishes a traceability among itemized software requirements specifications, design elements, code elements, and test cases (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It also supports various associated query, analysis, and report-generation capabilities.

requirements validation. (1) confirmation that requirements (individually and as a set) define the right system as
intended by the stakeholders (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.22) See also: requirements verification, software requirements validation

requirements verification. (1) confirmation by examination that requirements (individually and as a set) are well formed (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.23) Note: This means that a requirement or a set of requirements has been reviewed to ensure the characteristics of good requirements are achieved. See also: requirements validation, software requirements verification

reseller. (1) organization that purchases goods or services with an intention of selling them to another customer and possibly supporting them (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.29)

reserve. (1) a provision in the project management plan to mitigate cost and/or schedule risk. Often used with a modifier (e.g., management reserve, contingency reserve) to provide further detail on what types of risk are meant to be mitigated (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: buffer, contingency allowance See also: buffer

reserve analysis. (1) an analytical technique to determine the essential features and relationships of components in the project management plan to establish a reserve for the schedule duration, budget, estimated cost, or funds for a project. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

reserved word. (1) word in a programming language, of which the meaning is fixed by the rules of that language and which, in certain or all contexts, cannot be used by the programmer for any purpose other than its intended one (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

reset. (1) to set a variable, register, or other storage location back to a prescribed state (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: clear, initialize

residual control. (1) microprogramming technique in which the meaning of a field in a microinstruction depends on the value in an auxiliary register (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: bit steering, two-level encoding


resource. (1) asset that is utilized or consumed during the execution of a process (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.45) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.38) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 3.42) (2) team member or any physical item needed to complete the project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) role (with respect to that action) in which the enterprise object fulfilling the role is essential to the action, requires allocation, or can become unavailable (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.5) (4) any physical or virtual component of limited availability within a
computer system available for a given purpose and managed by the runtime platform (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM). 4) Note: Allocation of a resource can constrain other behaviors for which that resource is essential. Resources can be reusable, renewable or consumable. A consumable resource can become unavailable after some amount of use or after some amount of time (in case a duration or expiry has been specified for the resource).

resource allocation. (1) allocation (partitioning) of responsibility to different organizational units (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

resource breakdown structure (RBS). (1) a hierarchical representation of resources by category and type (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

resource calendar. (1) a calendar that identifies the working days and shifts on which each specific resource is available (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: project calendar

resource histogram. (1) a bar chart showing the amount of time that a resource is scheduled to work over a series of time periods (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: Resource availability may be depicted as a line for comparison purposes. Contrasting bars may show actual amounts of resource used as the project progresses.

resource leveling. (1) technique in which start and finish dates are adjusted based on resource constraints with the goal of balancing demand for resources with the available supply (ISO/IEC/IEEE 24765h:2019) (2) a resource optimization technique in which adjustments are made to the project schedule to optimize the allocation of resources and which may affect critical path (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: resource optimization technique, resource smoothing

resource management. (1) identification, estimation, allocation, and monitoring of the means used to develop a product or perform a service (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.1.4) Syn: RM

resource management plan. (1) a component of the project management plan that describes how project resources are acquired, allocated, monitored, and controlled (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

resource manager. (1) an individual with management authority over one or more resources (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

resource monitor task. (1) task ensuring sequential access to a resource (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

resource optimization techniques. (1) a technique in which activity start and finish dates are adjusted to balance demand for resources with the available supply (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: resource leveling, resource smoothing

resource requirement. (1) the types and quantities of resources required for each activity in a work package (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
resource-smoothing. (1) a resource optimization technique in which free and total float are used without affecting the critical path. See also: resource leveling, resource optimization technique.

resource utilization. (1) degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements. See also: resource leveling, resource optimization technique.

resource utilization measurement (RUM). (1) structure that provides information about resources associated with an IT asset in order to facilitate its management. Note: The RUM structure specifically contains information about the consumption of resources in relation to an IT asset.

resource-limited schedule. (1) project schedule whose schedule activity, scheduled start dates and scheduled finish dates reflect expected resource availability. See also: resource optimization technique.

respecialize. (1) change by an instance from being an instance of its current subclass to being an instance of one of the other subclasses in its current cluster. See also: specialize, unspecialize.

responding object. (1) object taking part in a communication, which is not the initiating object. See also: open distributed processing -- reference model: foundations, 13.4.2.

response time. (1) elapsed time between the end of an inquiry or command to an interactive computer system and the beginning of the system's response. See also: port-to-port time, think time, turnaround time.

responsibility. (1) generalization of properties (attributes, participant properties, and operations) and constraints. An assignment that can be delegated within a project management plan such that the assigned resource incurs a duty to perform the requirements of the assignment. An instance possesses knowledge, exhibits behavior, and obeys rules. These are collectively referred to as the instance's responsibilities. A class abstracts the responsibilities in common to its instances. A responsibility can apply to each instance of the class (instance-level) or to the class as a whole (class-level).

responsibility assignment matrix (RAM). (1) a grid that shows the project resources assigned to each work package. See also: responsibility assignment matrix.

responsive web design (RWD). (1) method for web page construction to detect the user's screen size and orientation and dynamically change the layout accordingly. See also: responsive web design.

responsiveness. (1) degree to which an IT service responds and provides outcomes in a prompt and timely way. See also: responsiveness.
(SQuaRE)--Service quality models, 3.2.6)

**restart.** (1) to cause a computer program to resume execution after a failure, using status and results recorded at a checkpoint (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**restart point.** (1) point in a computer program at which execution can be restarted following a failure (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: rescue point

**result.** (1) information returned to the client (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.13) (2) an output from performing project management processes and activities. Results include outcomes (e.g., integrated systems, revised process, restructured organization, tests, trained personnel, etc.) and documents (e.g., policies, plans, studies, procedures, specifications, reports, etc.) (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: product, deliverable


**retainage.** (1) portion of a contract payment that is withheld until contract completion to ensure full performance of the contract terms (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**retesting.** (1) re-execution of test cases that previously returned a "fail" result, to evaluate the effectiveness of intervening corrective actions (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.34) Syn: confirmation testing


**retirement phase.** (1) period of time in the software life cycle during which support for a software product is terminated (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software life cycle, system life cycle

**retrospective meeting.** (1) a team meeting at the end of an iterative cycle or at the end of a software project to reflect on what went well, what was learned, and what should be done differently next time (Software Extension to the PMBOK(R) Guide Fifth Edition)

**retrospective trace.** (1) trace produced from historical data recorded during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This differs from an ordinary trace, which is produced cumulatively during program execution. See also: execution trace, subroutine trace, symbolic trace, variable trace

**return.** (1) to transfer control from a software module to the module that called it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to assign a value to a parameter that is accessible by a calling module
(ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) (3) computer instruction or process that performs the transfer in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) See also: return code

**return code.** (1) code used to influence the execution of a calling module following a return from a called module (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary)

**return on investment (ROI).** (1) ratio of revenue from output (product or service) to development and production costs, which determines whether an organization benefits from performing an action to produce something (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary)

**return value.** (1) value assigned to a parameter by a called module for access by the calling module (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary)

**reusability.** (1) degree to which an asset can be used in more than one system, or in building other assets (ISO/IEC 25010:2011 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models, 4.2.7.2) See also: generality

**reusable.** (1) pertaining to a software module or other work product that can be used in more than one computer program or software system (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary)

**reusable product.** (1) system, software, or hardware product developed for one use but having other uses, or one developed specifically to be usable on multiple projects or in multiple roles on one project (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) Note: Each use can include all or part of the product and can involve its modification. This term can be applied to any software or system product (for example, requirements or architectures), not just to software or system itself.


**reused source statement.** (1) unmodified source statement obtained for the product from an external source (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary)

**revenue alternative.** (1) alternative that is described in terms of its complete cash-flow stream with expense and income (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) See also: service alternative

**reverse engineering.** (1) determining what existing software will do and how it is constructed (to make intelligent changes) (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) (2) a software engineering approach that derives a system's design or requirements from its code (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary)

**reversible execution.** (1) debugging technique in which a history of program execution is recorded and then replayed under the user's control, in either the forward or backward direction (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) Syn: backward execution, playback, replay, reverse execution

**review.** (1) process, which can include a meeting, in which work products are presented to some stakeholders for comment or approval (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) (2) process or

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.

Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
meeting during which a work product, or set of work products, is presented to project personnel, managers, users, customers, or other interested parties for comment or approval (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.52)

**review folder.** (1) entity for binding one or more related reviews, including a list of the reviews and information common to the reviews (ISO/IEC 23396:2020, Systems and software engineering Capabilities of review tools, 3.2) Note: The information common to the reviews can include information on members who can participate in or organize the reviews, and information on the classification given to the issues identified during the reviews.

**review/audit output.** (1) review or audit artifacts that are expected through conduct of the technical review or audit and that can be considered elements of exit criteria (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1) Syn: review output, audit output

**revocation.** (1) process of revoking an entitlement or entitlement schema (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.21) Note: An entitlement is sometimes revoked by the organization which originally issued it. The entitlement schema (Ent) enables the recording of entitlement revocations. Specific Ent transactions can also be revoked, e.g., to correct errors or record the rescinding of entitlement allocations.

**reward hacking.** (1) activity performed by an agent to maximize its reward function to the detriment of meeting the original objective (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 6.1.65)

**rework.** (1) action taken to bring a defective or nonconforming component into compliance with requirements or specifications. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**RFC.** (1) request for change (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2)

**RFI.** (1) request for information (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


**RFQ.** (1) request for quotation (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**right.** (1) privilege or benefit granted by a software entitlement (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.22)

Verification and Validation, 3.1) (3) combination of the likelihood of an abnormal event or failure and the consequence(s) of that event or failure to a system's components, operators, users, or environment (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (4) measure that combines both the likelihood that a system hazard will cause an accident and the severity of that accident (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.3) (5) uncertain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (6) combination of the probability of occurrence of harm and the severity of that harm (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.32) (7) function of the probability of occurrence of a given threat and the potential adverse consequences of that threat's occurrence (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.12) Note: Risk is often characterized by reference to potential events and consequences, or a combination of these. Risk is often expressed in terms of a combination of the consequences of an event and the associated likelihood of occurrence. Uncertainty is the state, even partial, of deficiency of information related to, understanding or knowledge of, an event, its consequence, or likelihood. The probability of occurrence includes the exposure to a hazardous situation, the occurrence of a hazardous event, and the possibility to avoid or limit the harm. See also: opportunity

risk analysis. (1) process of examining identified risk factors for probability of occurrence, potential loss, and potential risk-handling strategies (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

risk appetite. (1) the degree of uncertainty an entity is willing to accept, in anticipation of a reward (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

risk audit. (1) examination and documentation of the effectiveness of risk responses in dealing with identified risks and their root causes, as well as the effectiveness of the risk management process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) a type of audit used to consider the effectiveness of the risk management process (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

risk avoidance. (1) course of action that removes a risk factor from further consideration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) a risk response strategy whereby the project team acts to eliminate the threat or protect the project from its impact (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


risk categorization. (1) organization by sources of risk (e.g., using the RBS), the area of the project affected (e.g., using the WBS), or other useful category (e.g., project phase) to determine the areas of the project most exposed to the effects of uncertainty (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

risk category. (1) a group of potential causes of risk (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: A risk category is a characterization of a source of risk. See also: source


Note: Risk criteria are based on organizational objectives and external and internal context. Risk criteria can be derived from standards, laws, policies, and other requirements. Risk criteria can include associated cost and benefits, legal and statutory requirements, socio-economic and environmental aspects, the concerns of stakeholders, priorities and other inputs to the assessment.

**risk data quality assessment.** (1) technique to evaluate the degree to which the data about risks is useful for risk management (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**risk enhancement.** (1) a risk response strategy whereby the project team acts to increase the probability of occurrence or impact of an opportunity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**risk escalation.** (1) a risk response strategy whereby the team acknowledges that a risk is outside its sphere of influence and shifts the ownership of the risk to a higher level of the organization where it is more effectively managed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**risk exploiting.** (1) a risk response strategy whereby the project team acts to ensure that an opportunity occurs (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**risk exposure.** (1) potential loss presented to an individual, project, or organization by a risk (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.10) (2) an aggregate measure of the potential impact of all risks at any given point in time in a project, program, or portfolio (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: Risk exposure is commonly defined as the product of a probability and the magnitude of a consequence, that is, an expected value or expected exposure.

**risk factor.** (1) potential problem that would be detrimental to a planned activity, project, or program, characterized by the probability of problem occurrence (0 _ p _ 1) and a potential loss (of life, money, property, reputation, and so on) if the problem occurs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Both probability and potential loss might change over time.

**risk handling.** (1) course of action taken in response to a risk factor (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: includes risk acceptance, risk avoidance, risk transfer, and risk mitigation.

**risk identification.** (1) organized, systematic approach to determining the risk factors associated with a planned activity, project, or program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: identify risks

**risk leverage factor (rlf).** (1) rlf = (reb _ rea)/rmc, where reb is risk exposure before risk mitigation, rea is risk exposure after risk mitigation, and rmc is the risk mitigation activity's cost (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Larger rlfs indicate better mitigation strategies.

**risk management.** (1) organized process for identifying and handling risk factors (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) an organized means of identifying and measuring risk (risk assessment) and developing, selecting, and managing options (risk analysis) for resolving (risk handling) these risks. (3) organized, analytic process to identify what might cause harm or loss (identify risks); to assess and quantify the identified risk factors; to choose and implement risk responses (risk mitigation, risk acceptance, risk avoidance, risk transfer); and to monitor and control the risk. (4) organized means of identifying and measuring risk (risk assessment) and developing, selecting, and managing options (risk analysis) for resolving (risk handling) these risks. (5) organized, analytic process to identify what might cause harm or loss (identify risks); to assess and quantify the identified risk factors; to choose and implement risk responses (risk mitigation, risk acceptance, risk avoidance, risk transfer); and to monitor and control the risk. (6) organized, analytic process to identify what might cause harm or loss (identify risks); to assess and quantify the identified risk factors; to choose and implement risk responses (risk mitigation, risk acceptance, risk avoidance, risk transfer); and to monitor and control the risk.
risks; and to develop and, if needed, implement an appropriate approach to prevent or handle causes of risk that could result in significant harm or loss. (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary) Note: The primary goal of risk management is to identify and respond to potential problems with sufficient lead-time to avoid a crisis situation. Includes initial identification and handling of risk factors as well as continuous risk management.

**risk management plan.** (1) A component of the project, program or portfolio management plan that describes how risk management activities will be structured and performed. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: risk register

**risk metric.** (1) Objective measure associated with a risk factor to be mitigated. (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary)

**risk mitigation.** (1) A risk response strategy whereby the project team acts to reduce the probability of occurrence or impact of a risk. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: includes executing contingency plans when a risk metric crosses a predetermined threshold (when a risk becomes an issue or results in a problem)

**risk monitoring and control.** (1) Tracking identified risks, monitoring residual risks, identifying new risks, executing risk response plans, and evaluating their effectiveness throughout the project life cycle. (ISO/IEC/IEEE 24765:2017 Systems and software engineering—Vocabulary)


**risk profile.** (1) Description of any set of risks. (ISO/IEC/IEEE 16085:2021 Systems and software engineering—Life cycle processes—Risk management, 3.8) Note: The set of risks can contain those that relate to the whole organization, part of the organization, or one or more projects.

**risk reassessment.** (1) Identifying new risks, reassessing current risks, and closing risks that are outdated. (ISO/IEC/IEEE 24765h:2019)


**risk register.** (1) A repository in which outputs of risk management processes are recorded. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: The risk register details all identified risks, including description, category, cause, probability of occurring, impact(s) on objectives, proposed responses, owners, and current status. It can be kept in a database. See also: risk management plan

**risk report.** (1) A project document developed progressively throughout the Project Risk Management processes, which summarizes information on individual project risks and the level of overall project risk. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**risk review.** (1) A meeting to examine and document the effectiveness of risk responses in dealing with overall project
risk and with identified individual project risks (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

risk sharing. (1) a risk response strategy whereby the project team allocates ownership of an opportunity to a third party who is best able to capture the benefit of that opportunity (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


risk threshold. (1) measure of the level of uncertainty or the level of impact at which a stakeholder may have a specific interest. Below that risk threshold, the organization will accept the risk. Above that risk threshold, the organization will not tolerate the risk. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.9) (2) the level of risk exposure above which risks are addressed and below which risks may be accepted (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Note: Different risk thresholds can be defined for each risk, risk category or combination of risks, based on differing risk criteria. Below that risk threshold, the organization will accept the risk. Above that risk threshold, the organization will not tolerate the risk.

risk tolerance. (1) the degree, amount, or volume of risk that an organization or individual will withstand (ISO/IEC/IEEE 16085:2021 Systems and software engineering--Life cycle processes--Risk management, 3.10)

risk transfer. (1) transferring responsibility for managing a risk factor to another organization or functional entity better able to mitigate the risk factor (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

risk transference. (1) a risk response strategy whereby the project team shifts the impact of a threat to a third party, together with ownership of the response (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


risk trigger. (1) predetermined threshold value of a risk metric that triggers invocation of a contingency plan when the risk metric crosses the threshold (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

risk-based testing. (1) testing in which the management, selection, prioritization, and use of testing activities and resources are consciously based on corresponding types and levels of analyzed risk (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.35)

risk-free MARR. (1) MARR that has not been adjusted to address the risk in an alternative, because the risk is considered insignificant in the decision analysis (ISO/IEC/IEEE 24765:2017 Systems and software engineering-
Vocabulary

**RJE.** (1) remote job entry *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**RM.** (1) relationship manager *(ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.26)*


**roadmap.** (1) detailed plan to guide progress towards a goal *(ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.25)*

**robot.** (1) programmed actuated mechanism with a degree of autonomy, moving within its environment, to perform intended tasks *(ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.66)* *(Note: A robot includes the control system and interface of the control system. Robots can be classified into industrial robots or service robots, according to their intended application.)*

**robotics.** (1) techniques involved in designing, building, and using robots *(ISO/IEC 2382:2015 Information technology - -Vocabulary)*

**robustness.** (1) degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(See also: error tolerance, fault tolerance)*

**ROI.** (1) return on investment *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**role.** (1) participation of an entity in a relationship *(ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (2)* *(A defined function to be performed by a project team member, such as testing, filing, inspecting, coding. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3)* *(An expression of an object playing a part in a relationship *(ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.5)* *(4) Formal placeholder in the specification of a composite object that identifies those aspects of the behavior of some component object required for it to form part of the composite and links them as constraints on an actual object in an instance of the composite *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.17)*

**role name.** (1) name that more specifically names the nature of a related value class or state class *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.167)* *(2) Name assigned to a foreign key attribute to represent the use of the foreign key in the entity *(IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.167)* *(Note: For a relationship, a role name is a name given to a class in a relationship to clarify the participation of that class in the relationship, that is, connote the role played by a related instance. For an attribute, a role name is a name used to clarify the sense of the value class in the context of the class for which it is a property.)*
role-based reviewing. (1) technique where reviewers review a work product from the perspective of different stakeholder roles (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.16) Note: Typical stakeholder roles include specific user types, such as work product maintainer, tester, and developer.

role-playing session. (1) technique of engagement including the relevant stakeholders in an interactive simulation of the dynamic interactions that are part of the system design (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)

roll in. (1) to transfer data or computer program segments from auxiliary storage to main storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: roll out, swap

roll out. (1) to transfer data or computer program segments from main storage to auxiliary storage for the purpose of freeing main storage for other uses (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: roll in, swap

rolling wave planning. (1) an iterative planning technique in which the work to be accomplished in the near term is planned in detail, while the work in the future is planned at a higher level (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

ROM. (1) read-only memory (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

root arrow segment. (1) arrow segment of a junction from which other arrow segments branch or to which other arrow segments join (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.99) Syn: root, root segment

root cause. (1) source of a defect such that if it is removed, the defect is decreased or removed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

root cause analysis. (1) an analytical technique used to determine the basic underlying reason that causes a variance or a defect or a risk. A root cause may underlie more than one variance or defect or risk. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: root-cause analysis

root compiler. (1) compiler whose output is a machine independent, intermediate-level representation of a program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A root compiler, when combined with a code generator, comprises a full compiler.


routine. (1) subprogram that is called by other programs and subprograms (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) function or procedure invocable for a single purpose (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) program, or part of a program, that has some general or frequent use (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: The terms ‘routine,’ ‘subprogram,’ and ‘subroutine’ are defined and used differently in different programming languages. See also: coroutine, subroutine


RTC. (1) real-time clock (ISO/IEC/IEEE 24765d:2015)

RTM. (1) requirements traceability matrix (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)
RTOS. (1) real-time operating system (ISO/IEC/IEEE 24765c:2014)

rule. (1) constraint on a system specification (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 11.2.7) (2) single column through the condition and action entry parts of the decision table, defining a unique set of conditions to be satisfied and the actions to be taken in consequence (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.4) Note: A rule is satisfied if all conditions meet the condition entries of the rule.

Rule and Constraint Language (RCL). (1) declarative specification language that is used to express the realization of responsibilities and to state queries (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.168) See also: specification language

rule-based language. (1) nonprocedural language that permits the user to state a set of rules and to express queries or problems that use these rules (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: declarative language, interactive language

RUM. (1) resource utilization measure (ISO/IEC 19770-4:2017 Information technology -- IT asset management -- Part 4: Resource utilization measurement, 3.8)

RUM creator. (1) entity that initially creates a RUM (ISO/IEC 19770-4:2017 Information technology -- IT asset management -- Part 4: Resource utilization measurement, 3.9) Syn: resource utilization measurement creator

run. (1) in software engineering, a single, usually continuous, execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to execute a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: run time

run time. (1) instant at which a computer program begins to execute (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) period of time during which a computer program is executing (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) stage that a member product is executed (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.11) Syn: running time See also: execution time

runtime platform. (1) set of hardware and software components that implement the services utilized by the application software (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)

RUR. (1) Reference User Requirements (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 4)

RUSP. (1) ready-to-use software product (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.6)

RWD. (1) responsive web design (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

SaaS. (1) software as a service (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1)

SAD. (1) software architecture description (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on
safety. (1) expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.48) (2) freedom from unacceptable risk (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.14) Note: See ISO/IEC Guide 51 for issues related to safety.
safety criteria. (1) limits of acceptable risk associated with a hazard (ISO/IEC TS 15504-10:2011 Information technology--Process assessment--Part 10: Safety extension, 3.4) Note: These limits can be defined as imposed safety targets or developed from analysis or development policy.
safety demonstration. (1) body of evidence and rationale that shows an item is justified as being safe within allowed limits on risk (ISO/IEC TS 15504-10:2011 Information technology--Process assessment--Part 10: Safety extension, 3.3)
safety integrity requirement. (1) likelihood of a safety-related system satisfactorily performing the required safety functions under stated conditions (ISO/IEC TS 15504-10:2011 Information technology--Process assessment--Part 10: Safety extension, 3.6)
safety life cycle. (1) project or product life cycle in which safety processes are performed (ISO/IEC TS 15504-10:2011 Information technology--Process assessment--Part 10: Safety extension, 3.7)
safety note. (1) safety-related information that is collected or grouped in a document or section of a document in a meaningful organizational system to explain safety measures, raise safety awareness, and provide a basis for safety-related training of persons (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.33)
safety requirement. (1) requirement that is needed to ensure the safety of the product (ISO/IEC TS 15504-10:2011 Information technology--Process assessment--Part 10: Safety extension, 3.8)
safety sign. (1) sign giving a general safety message, obtained by a combination of color and geometric shape and which, by the addition of a graphical symbol, gives a particular safety message (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.34)
safety validation. (1) assurance, based on examination and tests, that the safety goals are sufficient and have been achieved (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.32)
safety-critical software. (1) software that falls into one or more of the following categories: a) software whose inadvertent response to stimuli, failure to respond when required, response out-of-sequence, or response in combination with other responses can result in an accident b) software that is intended to mitigate the result of an accident c) software that is intended to recover from the result of an accident (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.4)
safety-critical system. (1) system whose failure or malfunction can result in one (or more) of the following outcomes: death or serious injury to people, loss or severe damage to equipment or property, or environmental harm (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools,
3.15) Note: Software engineering for safety-critical systems emphasizes (1) process engineering and management; (2) selecting the appropriate tools and environment for the system; (3) adherence to requirements. Syn: life-critical system

SAINV. (1) schedule as independent variable (Software Extension to the PMBOK(R) Guide Fifth Edition)

SAM. (1) software asset management (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.2)

SAM owner. (1) individual at a senior organization-wide level who is identified as being responsible for SAM (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.11) (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.30)

SAM practitioner. (1) individual involved in the practice or role of managing software assets (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.31) Note: A SAM practitioner is often involved in the collection or reconciliation of software inventory and software entitlements.

SAM program scope. (1) clear statement listing of all parts of the organization and types of software, assets, and platforms covered by a SAM program (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.32)

SAM tool. (1) software used to assist in and automate parts of the process of management of software assets (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.25) Syn: software asset management tool

SAMP. (1) strategic asset management plan (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.53)

sample instance diagram. (1) form of presenting example instances in which instances are shown as separate graphic objects (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.169) Note: The graphic presentation of instances can be useful when only a few instances are presented. See also: sample instance table

sample instance table. (1) form of presenting example instances in which instances are shown as a tabular presentation (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.170) Note: The tabular presentation of instances can be useful when several instances of one class are to be presented. See also: sample instance diagram

SAR. (1) software requirements and architecture review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2) See also: SRR


satisfaction. (1) freedom from discomfort and positive attitudes towards the use of the product (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQaRE) -- Common Industry Format (CIF) for usability test reports, 4.4) (2) degree to which user needs are satisfied when a product or system is used in a specified context of use (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality
satisficing. (1) decision technique that discards any alternative with an attribute value outside an acceptable range

SBI. (1) serial bus interface (ISO/IEC/IEEE 24765e:2015)

SBS. (1) system breakdown structure (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 4.2)

scaffolding. (1) computer programs and data files built to support software development and testing, but not intended to be included in the final product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: programming support environment

scalar. (1) value that is atomic (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.171) Note: That is, having no parts. See also: collection-valued

scalar-valued class. (1) class in which each instance is a single value (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.173) See also: collection-valued class


scale. (1) ordered set of values, continuous or discrete, or a set of categories to which the attribute is mapped (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.30) (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.34) Note: The type of scale depends on the nature of the relationship between values on the scale. Metrics using nominal or ordinal scales produce qualitative data, and metrics using interval and ratio scales produce quantitative data.

scatter diagram. (1) correlation chart with two axes that plots multiple observations of an independent and a dependent variable, resulting in a regression line that defines the relationship of the variables. (ISO/IEC/IEEE 24765h:2019)


scenario. (1) step-by-step description of a series of events that occur concurrently or sequentially (ISO/IEC/IEEE 24765e:2015) Note: A scenario can be a user story, use case, operational concept, or sequence of events the software could encounter. Syn: script See also: use case

scenario testing. (1) class of test design technique in which tests are designed to execute individual scenarios (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.36)

scenario-based reviewing. (1) technique where the review is guided by determining the ability of the work product
to address specific scenarios (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.17)

schedule as independent variable (SAIV). (1) a date-certain scheduling method for a project with a specific end date, after which the value of the project declines precipitously or a penalty for non-completion is applied (Software Extension to the PMBOK(R) Guide Fifth Edition)

schedule baseline. (1) the approved version of a schedule model that can be changed only through formal change control procedures and is used as a basis for comparison to actual results (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

schedule compression. (1) technique used to shorten the schedule duration without reducing the project scope (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: crashing, fast tracking

schedule data. (1) the collection of information for describing and controlling the schedule (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

schedule development. (1) process of creating the project schedule by analyzing activity sequences, activity durations, resource requirements, and schedule constraints (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

schedule forecast. (1) estimate or prediction of conditions and events in the project's future based on information and knowledge available at the time the schedule is calculated (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

schedule management plan. (1) a component of the project management plan that establishes the criteria and the activities for developing, monitoring, and controlling the schedule (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

schedule model. (1) a representation of the plan for executing the project's activities, including durations, dependencies and other planning information, used to produce a project schedule along with other scheduling artifacts (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

schedule network analysis. (1) a technique of identifying early and late start dates, as well as early and late finish dates, for the uncompleted portions of project schedule activities (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: network analysis, schedule analysis See also: backward pass, critical path method, critical chain method, resource leveling

schedule performance index (SPI). (1) a measure of schedule efficiency expressed as the ratio of earned value to planned value (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

schedule variance (SV). (1) a measure of schedule performance expressed as the difference between the earned value and the planned value (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

scheduler. (1) computer program, usually part of an operating system, that schedules, initiates, and terminates jobs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

scheduling tool. (1) tool that provides schedule component names, definitions, structural relationships, and formats that support the application of a scheduling method (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

SCM. (1) software configuration management (ISO/IEC/IEEE 24765:i:2020)
SCMP. (1) software configuration management plan (ISO/IEC/IEEE 24765d:2015)

SCMS. (1) Source Control Content Management System (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 5)

SCN. (1) specification change notice (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

scope. (1) the sum of the products, services, and results to be provided as a project (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) behavior that a system is expected to exhibit (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.1.1) Note: A scope change almost always requires an adjustment to the project cost or schedule. See also: project scope, product scope, scope of the FSM

scope of the FSM. (1) the set of functional user requirements to be included in a specific FSM instance (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.11) (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FISMA 1.1 functional size measurement method, A.12) (2) set of functional user requirements to be included in a specific functional size measurement instance (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.24) Note: The Scope of the FSM is determined by the purpose for measuring the software.

screen capture. (1) representation of what the user will see while using the software (ISO/IEC/IEEE 24765a:2011) Syn: screen dump

scripted testing. (1) dynamic testing in which the tester's actions are prescribed by written instructions in a test case (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.37) (2) testing performed based on a documented test script (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.23) Note: normally applies to manually executed testing, rather than the execution of an automated script

SCRM. (1) supply chain risk management (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

scrum. (1) iterative project management framework used in agile development, in which a team agrees on development items from a requirements backlog and produces them within a short duration of a few weeks (ISO/IEC/IEEE 24765h:2019)

scrum master. (This term is deprecated.) (1) person who facilitates the scrum process within a team or project (ISO/IEC/IEEE 24765h:2019) Syn: scrum leader

scrum meeting. (1) brief daily project status meeting or other planning meeting in agile development methodologies (ISO/IEC/IEEE 24765h:2019) Note: The scrum meeting is usually chaired by the scrum master.

scrum report. (1) report that documents the daily activities of a scrum team, recording any problems or issues to be
dealt with (ISO/IEC/IEEE 24765h:2019)

scrum team. (1) members of an agile development team working together under the scrum process, usually led by the scrum leader and project owner (ISO/IEC/IEEE 24765h:2019)

SCU. (1) SNAP (software non-functional assessment process) counting unit (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)

SDa. (1) submit data activity (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)


SDP. (1) software development plan (ISO/IEC/IEEE 16326:2019 Systems and software engineering -- Life cycle processes -- Project management, 3)

SDR. (1) system design review (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

SDRAM. (1) synchronous dynamic random access memory (ISO/IEC/IEEE 24765c:2014)


search algorithm. (1) algorithm that systematically visits a subset of all possible states (or structures) until the goal state (or structure) is reached (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.68)


second normal form. (1) result of a normalization process that transforms groups of data so that each non-key attribute depends on the key attribute(s) of the group of data and all parts of the key attribute(s) (ISO/IEC/IEEE 24765a:2011)

secondary data. (1) data that has already been collected from various sources, such as governmental census or statistical bureaus, nonprofits, academic institutions and consultancies (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)


secondary user. (1) user who interacts with the product to support the primary users (ISO/IEC 25030:2019 Systems and software engineering--Systems and software quality requirements and evaluation (SQuaRE)--Quality requirements framework, 3.17) See also: operator

secondary window. (1) window that contains information that depends on information in another window (the primary window) (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.43)

secure hash algorithm (SHA). (1) algorithm that is used to verify data integrity through the creation of a message
digest from data input
(which may be a message of any length) (ISO/IEC 19770-5:2015 Information technology-- IT asset management-- Overview and vocabulary, 3.33) Note: includes SHA-1 (160-bit digest) and SHA-2 (224- to 512-bit digest)

security. (1) protection against intentional subversion or forced failure (ISO/IEC/IEEE 15288:2015 Systems and software engineering-- System life cycle processes, 4.1.41) (2) defining, achieving, and maintaining confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of a system (ISO/IEC/IEEE 24765e:2015) (3) degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization (ISO/IEC 25010:2011 Systems and software engineering-- Systems and software Quality Requirements and Evaluation (SQuaRE)-- System and software quality models, 4.2.6) (4) protection of computer hardware or software from accidental or malicious access, use, modification, destruction, or disclosure (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (5) protection of information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them (ISO/IEC/IEEE 12207:2017 Systems and software engineering-- Software life cycle processes, 4.39) (6) protection against intentional subversion or forced failure, containing a composite of four attributes: confidentiality, integrity, availability and accountability, plus aspects of a fifth, usability, all of which have the related issue of their assurance (ISO/IEC/IEEE 12207:2017 Systems and software engineering-- Software life cycle processes, 3.1.49) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering-- Lifecycle management-- Part 1: Guidelines for life cycle management, 2.45) (7) degree to which an IT service protects both users' assets and access to their information so that users have the degree of information access appropriate to their levels of authorization (ISO/IEC TS 25011:2017 Information technology-- Systems and software Quality Requirements and Evaluation (SQuaRE)-- Service quality models, 3.2.3) (8) resistance to intentional, unauthorized acts designed to cause harm or damage to a system (ISO/IEC 23643:2020, Software and systems engineering-- Capabilities of software safety and security verification tools, 3.16) Note: A composite of four attributes: confidentiality, integrity, availability, and accountability, plus aspects of a fifth: usability, all of which have the related issue of their assurance. Security pertains to personnel, data, communications, and the physical protection of computer installations.

security accreditation. (1) formal declaration by management that an IT system is approved to operate in a particular security mode using a prescribed set of safeguards at an acceptable level of risk (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) independent accreditation body's certification that an IT system meets a predetermined security standard (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

security and intellectual property scheme. (1) established and operated management system in the entity to ensure the security and intellectual property of its information items (ISO/IEC TR 29110-5-1-3:2017 Systems and software engineering-- Lifecycle profiles for Very Small Entities (VSEs)-- Part 5-1-3: Software engineering -- Management and engineering guide: Generic profile group-- Intermediate profile, 3.5)

security authority. (1) administrator responsible for the implementation of a security policy (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 15.1.2)

security branch. (1) branch, created at the time of a release, to which only security commits are made (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
security domain. (1) domain in which the members are obliged to follow a security policy established and administered by a security authority (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 15.1.3) Note: The security authority is the controlling object for the security domain.

security interaction policy. (1) those aspects of the security policies of different security domains that are necessary in order for interactions to take place between those domains (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 15.1.4)

security kernel. (1) small, self-contained collection of key security-related statements that works as a privileged part of an operating system, specifying and enforcing criteria that must be met for programs and data to be accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

security level. (1) combination of a hierarchical security classification and a security category that represents the sensitivity of an object or the security clearance of an individual (ISO/IEC 23643:2020, Software and systems engineering-Capabilities of software safety and security verification tools, 3.17) Note: The minimum security level is an indication of the minimum protection required.

security policy. (1) rules for need-to-know and access-to-information at each project organization level (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) set of rules that constrains one or more sets of activities of one or more sets of objects (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 15.1.1)

security testing. (1) type of testing conducted to evaluate the degree to which a test item, and associated data and information, are protected so that unauthorized persons or systems cannot use, read, or modify them, and authorized persons or systems are not denied access to them (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.38)


SEE service. (1) one or more service operations to support life cycle activities for the SEE (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.8) Note: A SEE Service supplier provides a SEE Service for a SEE Service acquirer.

SEE service acquirer. (1) actor that acquires an SEE service (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.12)

SEE service supplier. (1) actor that supplies an SEE service (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.13)

segment. (1) one of the subsystems or combinations of subsystems that make up an overall system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) in storage allocation, a self-contained portion of a computer program that can be executed without maintaining the entire program in main storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) collection of data that is stored or transferred as a unit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) in path analysis, a sequence of computer program statements between two consecutive branch points (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (5) to divide a system, computer program, or data file into segments as in (1), (2), or (3)
segmented executor. (1) set of physically distinct artifacts, a physical partition of the executor (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: Each segment encapsulates independent state and is capable of being independently activated. Each segment provides at least one facet.


selective dump. (1) a dump of designated storage location areas only (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: change dump, dynamic dump, memory dump, postmortem dump, snapshot dump, static dump

selective trace. (1) variable trace that involves only selected variables (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

self-assembly product. (1) product that is intended to be assembled or erected from separate components without the help of a supervisor or trainer and relying only on step-by-step instructions (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.35) Note: includes both products supplied in the form of a kit of component parts and products that incorporate structural elements requiring folding, locking or tensioning (e.g. child pushchairs or camping equipment). Syn: flatpack product, ready-to-assemble product

self-contained. (1) process in which no prior or subsequent processing steps are needed to initiate or complete the functional user requirements (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.47)

self-descriptiveness. (1) the degree to which a system or component contains enough information to explain its objectives and properties (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) software attributes that explain a function's implementation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: maintainability, testability, usability

self-documented. (1) pertaining to source code that contains comments explaining its objectives, operation, and other information useful in understanding and maintaining the code (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

self-learning system. (1) adaptive system that changes its behavior based on learning from the practice of trial and error (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.69)

self-organizing team. (1) a team formation where the team functions with an absence of centralized control (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)
self-relative address. (1) address that must be added to the address of the instruction in which it appears to obtain the address of the storage location to be accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) See also: base address, indexed address, offset, relative address

seller. (1) a provider or supplier of products, services, or results to an organization. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: supplier

seller proposals. (1) formal responses from sellers to a request for proposal or other procurement document specifying the price, commercial terms of sale, and technical specifications or capabilities the seller will do for the requesting organization that, if accepted, would bind the seller to perform the resulting agreement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


semantic error. (1) error resulting from a misunderstanding of the relationship of symbols or groups of symbols to their meanings in a given language (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) See also: syntactic error

semantic label. (1) tag that describes the content rather than its format (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.29)

semantics. (1) meaning of the syntactic components of a language (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.175) (2) relationships of symbols or groups of symbols to their meanings in a given language (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) See also: syntax

semaphore. (1) shared variable used to synchronize concurrent processes by indicating whether an action has been completed or an event has occurred (ISO/IEC/IEEE 24765:2017 Systems and software engineering - Vocabulary) See also: flag, indicator

SEMDM. (1) software engineering metamodel for development methodologies (ISO/IEC 24744:2014 Software Engineering--Metamodel for development methodologies, 4.2)

semi-formal verification. (1) method based on a description given in semi-formal notation (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.18)

semiconductor. (1) substance with conductive properties between those of a conductor and an insulator (ISO/IEC/IEEE 24765c:2014)


sensitivity analysis. (1) an analysis technique to determine which individual project risks or other sources of uncertainty have the most potential impact on project outcomes, by correlating variations in project outcomes with variations in elements of a quantitative risk analysis model (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) risk-analysis technique that studies how changes in the values of estimated parameters affects the desirability of an alternative. Parameters where small changes in estimated values cause larger
changes in desirability are said to be more sensitive. Sensitivity analysis guides the decision maker in identifying the estimated parameters (the sensitive ones) that deserve more careful study to improve the accuracy of the estimate. *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: It examines the extent to which the uncertainty of each project element affects the objective being examined when all other uncertain elements are held at their baseline values. The typical display of results is in the form of a tornado diagram.

**sentence.** *(1)* linguistic construct containing one or more terms and predicates *(ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 7.2)* Note: A sentence can express a proposition about the entities to which the terms refer. A predicate in a sentence can refer to a relationship between the entities referred to by the terms it links.


**separate documentation.** *(1)* documentation that can be used independently of the software *(ISO/IEC/IEEE 24765g:2018)* See also: embedded documentation

**sequence activities.** *(1)* the process of identifying and documenting relationships among the project activities *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**sequence diagram.** *(1)* a Unified Modeling Language (UML) diagram that depicts time-sequential ordering of interactions, as in a use case scenario of interactions between an actor and some system elements. Can be used to depict sequential and concurrent data flow or process flow. *(Software Extension to the PMBOK(R) Guide Fifth Edition)*

**sequential.** *(1)* pertaining to the occurrence of two or more events or activities in such a manner that one must finish before the next begins *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: consecutive, serial

**sequential clustering.** *(1)* task-structuring criterion in which objects that are constrained to execute sequentially are mapped to a task *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**sequential cohesion.** *(1)* type of cohesion in which the output of one task performed by a software module serves as input to another task performed by the module *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: coincidental cohesion, communicational cohesion, functional cohesion, logical cohesion, procedural cohesion, temporal cohesion

**serial.** *(1)* pertaining to the sequential transfer, occurrence, or processing of the individual parts of a whole, such as the bits of a character, using the same facilities for successive parts *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: parallel *(1), sequential*

**serial bus.** *(1)* shared channel that transmits data sequentially, bit-by-bit *(ISO/IEC/IEEE 24765e:2015)*

**serial bus interface (SBI).** *(1)* connection for bidirectional serial data communication *(ISO/IEC/IEEE 24765e:2015)*

Note: based on the IEEE 1394 standards.

**serial communication interface (SCI).** *(1)* unit that enables the serial exchange of data between a...

**serial construct.** (1) program construct consisting of a sequence of steps not involving a decision or loop


**serial expansion interface (SEI).** (1) connection to serve multiple serial channels (ISO/IEC/IEEE 24765e:2015)

**serial peripheral interface (SPI).** (1) synchronous (full duplex) serial communication interface used for two devices in embedded systems (ISO/IEC/IEEE 24765d:2015)

**server.** (1) hardware system or software program which provides a service to clients (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process implementing one or more operations on one or more objects (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.14)

**server object.** (1) object which performs some service on behalf of a client object (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.4.6)

**server-side.** (1) node, cluster or capsule, which: a) contains, or is potentially capable of containing, a basic engineering object that corresponds to a computational server object and stub, binder and protocol objects in a channel supporting operations involving the server object; or b) contains, or is a potentially capable of containing, a protocol object which (possibly via interactions with other engineering objects) can return a reply identifying another server-side (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.10)


**service alternative.** (1) alternative that is assumed to provide equivalent service to another alternative over their lives; all the revenue cash flows are ignored to simplify the comparison and only the expense cash flows are shown (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: revenue alternative

**service catalogue.** (1) documented information about services that an organization provides to its customers (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.29) Syn: service catalog

**service change request.** (1) formal procedure for submitting a request for an adjustment of a service (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.29) Syn: service change request
service component. (1) single unit of a service that when combined with other units will deliver a complete service (ISO/IEC/IEEE 24765c:2014) (2) CORBA component with behavior, no state, and no identity (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1) Note: A service component can consist of one or more configuration items.

service continuity. (1) capability to deliver a service without interruption, or with consistent availability as scheduled and agreed (ISO/IEC 24765c:2014)

service delivery policy. (1) formal, brief, and high-level statement that embraces an organization’s general beliefs, ethics, goals, and objectives of service(s) (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.31)

service delivery profile. (1) profile targeted at very small enterprises (VSEs) that need to perform and manage service delivery processes, either for systems or software products that they have developed or that were developed by others (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.54)

service design. (1) creation of a service solution(s); typically including the components which create the desired functionality, technology architecture that supports the components, the processes to support and manage the solution, the associated measures (internal performance or customer agreed measures), and the supply chain interfaces (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.32)

service desk. (1) customer-facing support group for centralized resolution of incidents, change requests, and complaints concerning a service (ISO/IEC/IEEE 24765a:2011)

service export. (1) interaction with the trading function in which a service offer is advertised, by adding the service offer to an identified set of service offers (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 14.5.1.2)

service import. (1) interaction with the trading function which searches an identified set of service offers to discover interfaces at which a service satisfying a specified type is available (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 14.5.1.3)

service level agreement (SLA). (1) documented agreement between the service provider and user that identifies services and service targets (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.8) (2) contract between a service provider (either internal or external) and the end user that defines the level of service expected from the service provider (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) documented agreement between the IT service provider and the user that identifies services and their agreed performance (ISO/IEC TS 25025:2021, Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Measurement of IT service quality,


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
3.7) **Note:** A service level agreement can also be established between the service provider and a supplier, an internal group, or a customer acting as a supplier. A service level agreement can be included in a contract or another type of documented agreement. An Application Management organization can be the service provider, but it can also be a customer itself, of another supplier.

**service management.** (1) set of capabilities and processes to direct and control the service provider's activities and resources for the design, transition, delivery and improvement of services to fulfill the service requirements (ISO/IEC/IEEE 24765c:2014)

**service management system (SMS).** (1) management system to direct and control the provision of one or more services (ISO/IEC/IEEE 24765c:2014) **Note:** The SMS includes all service management policies, objectives, plans, processes, documentation and resources required for the design, transition, delivery and improvement of services and to fulfill the requirements.

**service manager (SM).** (1) role that directly oversees the delivery of services and provides leadership and direction, has decision-making authority on all activities, and is a direct report or peer to the highest level of the organization (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecyle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.34) **Note:** The service manager may have more than one role in the delivery of services (such as assigning the responsibilities of the Control Manager and Service Manager to the same individual).

**service offer.** (1) information describing an interface, how to bind to it, and the service that can be invoked using it (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 14.5.1.1)

**service primitive.** (1) abstract definition of an interaction of channel objects that causes protocol exchanges between the protocol objects in the channel (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.11)

**service provider.** (1) organization or part of an organization that manages and delivers a service or services to the customer (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.7) (2) organization that manages and delivers a service or services to customers (ISO/IEC TS 25025:2021, Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Measurement of IT service quality, 3.6) **Note:** A customer can be internal or external to the service provider's organization.

**service provision system.** (1) system to provide IT service to users, including people, processes, technology, facilities and information (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.6)

**service request.** (1) request for information, or for a routine change or procedure with previously evaluated risk (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.25)

**session component.** (1) CORBA component with behavior, transient state, and identity that is not persistent (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)
set. (1) collection class with no duplicate members and where order is irrelevant (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.176) See also: bag, list

set-up time. (1) period of time during which a system or component is being prepared for a specific operation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: busy time, down time, idle time, setup time, up time

SETA. (1) systems engineering and technical assistance (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

SETR. (1) systems engineering technical review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)


SGML. (1) Standard Generalized Markup Language (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

SGRAM. (1) synchronous graphics random access memory (ISO/IEC/IEEE 24765c:2014)

SHA. (1) secure hash algorithm (ISO/IEC 19770-2:2015 (corr 2017), Information technology -- Software asset management -- Part 2: Software identification tag, 3.2)

shadow class. (1) class presented in a view that is specified in some other view (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.177)

shell. (1) computer program or routine that provides an interface between the user and a computer system or program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

should-cost estimate. (1) estimate of the cost of a product or service used to evaluate the reasonableness of a supplier's proposed price (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

shrink small outline package (SSOP). (1) thinner rectangular surface mount integrated circuit unit with gull-wing leads on the two long sides (ISO/IEC/IEEE 24765c:2014)

sign change coverage. (1) proportion of neurons activated with both positive and negative activation values divided by the total number of neurons in the neural network (normally expressed as a percentage) for a set of tests (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.70)

sign-sign coverage. (1) coverage level 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 stay the same (i.e. they do not change sign) (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.71)

signal. (1) atomic shared action resulting in one-way communication from an initiating object to a responding object (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.1)

(2) variation of a physical quantity used to represent data (ISO/IEC 2382:2015 Information technology -- Vocabulary)
A signal is an interaction.

**signal interface.** (1) Interface in which all the interactions are signals (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.6)

**signal interface signature.** (1) Interface signature for a signal interface (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 7.1.11) Note: A signal interface signature comprises a finite set of action templates, one for each signal type in the interface. Each action template comprises the name for the signal, the number, names and types of its parameters, and an indication of causality (initiating or responding, but not both) with respect to the object which instantiates the template.

**signature.** (1) Definition of the parameters of a given operation, including their number order, data types, and passing mode; the results if any; and the possible outcomes (normal vs. exceptional) that might occur (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.15) (2) Mathematical structure comprising a set of sorts and a set of operators (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.32) (3) Statement of what the interface to a responsibility &quot;looks like.&quot; (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.178) Note: A signature consists of the responsibility name, along with a property operator and the number and type of its arguments, if any. A type (class) can be specified for each argument to limit the argument values to being instances of that class.

**signpost.** (1) Text, symbol, or small graphic that helps the user identify where particular types of information are located or where the information in the current display fits into the whole document (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.45) Note: Information of different types can be indicated by symbols or graphics of different types.

**SIL.** (1) System integration laboratory (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)


**simple buffering.** (1) Buffering technique in which a buffer is allocated to a computer program for the duration of the program's execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dynamic buffering

**simplex receptacle.** (1) Specialization of a receptacle that only allows a single connection at a given time (ISO/IEC 19500-3:2012 Information technology--Object Management Group--Common Architecture Request Broker Architecture (CORBA)--Part 3: Components, 4.1)

**simplicity.** (1) Degree to which a system or component has a design and implementation that is straightforward and easy to understand (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) Software attributes that provide implementation of functions in the most understandable manner (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: complexity

**simulation.** (1) Model that behaves or operates like a given system when provided a set of controlled inputs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) Process of developing or using a model
as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) use of a data processing system to represent selected behavioral characteristics of a physical or abstract system (ISO/IEC 2382:2015 Information technology -- Vocabulary) (4) A simulation uses a project model that translates the uncertainties specified at a detailed level into their potential impact on objectives that are expressed at the level of the total project. Project simulations use computer models and estimates of risk, usually expressed as a probability distribution of possible costs or durations at a detailed work level, and are typically performed using Monte Carlo analysis. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: emulation

simulator. (1) device, computer program, or system that behaves or operates like a given system when provided a set of controlled inputs (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.72) See also: emulator

simultaneous. (1) pertaining to the occurrence of two or more events at the same instant of time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: concurrent

simultaneous recursion. (1) situation in which two software modules call each other (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

single boot. (1) having only one boot mode to start a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dual boot

single instance approach. (1) case where each method of delivery of same functionality is counted only once (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) See also: multiple instance approach

single-hit decision table. (1) decision table where any set of conditions will be satisfied by one, and only one, rule (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.2)

single-level encoding. (1) microprogramming technique in which different microoperations are encoded as different values in the same field of a microinstruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: single level encoding See also: two-level encoding


SIP. (1) system integration plan (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)

site map. (1) textual or graphical overview of the navigation structure of a website (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.23)

sizing. (1) process of estimating the amount of computer storage or the number of source lines required for a software system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: timing

skeleton. (1) object-interface-specific ORB component which assists an object adapter in passing requests to

skilled person. (1) individual with relevant technical education, training, or experience to enable perceiving risks and avoiding hazards occurring during use of a product (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.36) See also: competent person

sku. (1) stock keeping unit (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.48)

SLA. (1) service level agreement (ISO/IEC/IEEE 24765c:2014)

SLA compliance. (1) degree to which the service complies with the SLA (3.3.8) requirements (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.3.9) Syn: service level agreement compliance


SLI. (1) service level indicator (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

SLO. (1) service level objective (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

SLOC. (1) Source Lines of Code, the number of lines of programming language code in a program before compilation (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

slush. (1) preparation for a feature or code freeze (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: During this period, developers will commit code they have been working on but are discouraged from starting on new elements. If a freeze lasts for a long time, a slush might be introduced to ease its passing by allowing in some extra elements.

SM. (1) service manager (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.34)

Small and Medium Enterprise (SME). (1) enterprises which employ fewer than 250 persons and which have an annual turnover not exceeding 50 million euro, and/or an annual balance sheet total not exceeding 43 million euro (ISO/IEC TR 29110-5-6:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.6) (2) enterprise which employs fewer than 250 persons (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.52) (3) enterprise with fewer than 250 persons employed (ISO/IEC 29110-4-2:2021, Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 4-2: Software engineering: Profile specifications: Organizational management profile group, 3.1.5) Syn: small and medium-sized enterprise


small outline package (SOP). (1) rectangular surface mount integrated circuit unit with gull-wing leads on the two
SMS. (1) service management system (ISO/IEC/IEEE 24765i:2020)
SN. (1) symmetric net (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.2.7)
SNAP. (1) software non-functional assessment process (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)
snapshot dump. (1) dynamic dump of the contents of one or more specified storage areas (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: change dump, dynamic dump, memory dump, postmortem dump, selective dump, static dump
SOA. (1) service-oriented architecture (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)
soft copy image. (1) nonpermanent output of information in audio or visual format (ISO/IEC 2382:2015 Information technology --Vocabulary)
soft failure. (1) failure that permits continued operation of a system with partial operational capability (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: hard failure
software. (1) computer programs, procedures and possibly associated documentation and data pertaining to the operation of a computer system (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (2) all or a part of the programs, procedures, rules, and associated documentation of an information
processing system (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.34) (3) program or set of programs used to run a computer (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.46) (4) all or part of the programs which process or support the processing of digital information (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.49) (5) part of a product that is the computer program or the set of computer programs (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.34) Note: Software excludes information per se, such as the content of documents, audio and video recordings, graphics, and databases. Digital information which is managed by executable software (e.g. the content of documents and databases) is not considered software, even though program execution may depend on data values. Syn: SW See also: application software

software acquisition process. (1) period of time that begins with the decision to acquire a software product and ends when the product is no longer available for use (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software agent. (1) digital entity that perceives its environment and takes actions that maximize its chance of successfully achieving its goals (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.73)


software as a service (SaaS). (1) access to resources from client devices through thin client interface or program interface (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: The level of control over the resource provided to the consumer can vary with the service provider. In DevOps, SaaS can be automated as part of the DevOps pipeline. See also: IaaS, PaaS

software asset. (1) software that has potential or actual value to an organization (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.50) (2) description of a partial solution (such as a component or design document) or knowledge (such as requirements database or test procedures) that engineers use to build or modify software products (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4)

software asset management (SAM). (1) control and protection of software and related assets within an organization, and control and protection of information about related assets which are needed in order to control and protect software assets (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.35) (2) coordinated activity of an organization to realize value from software assets (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.51) Note: infrastructure and processes necessary for the effective management, control and protection of the software assets within an organization, throughout all stages of their lifecycle
software baseline. (1) set (one or more) of software configuration items formally designated and fixed at a specific time during the software life cycle (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

software characteristic. (1) inherent, possibly accidental, trait, quality, or property of software (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software component (SC). (1) entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.36) (2) functionally or logically distinct part of a software configuration item (SCI), distinguished for the purpose of convenience in designing and specifying a complex SCI as an assembly of subordinate elements (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) software system or element (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.57) Note: Software component refers to a part of a whole, such as a component of a software product or a component of a software identification tag.

software configuration item (SCI). (1) software entity that has been established as a configuration item (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: The SCI exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the software has been established as a configurable item. See also: computer software component, computer software configuration item, hardware configuration item, software item

software configuration management (SCM). (1) process of applying configuration management throughout the software life cycle to ensure the completeness and correctness of software configuration items (SCI) (ISO/IEC/IEEE 24765:2020)

software consumer. (1) entity that uses an entitlement of a software package (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.37)

software creator. (1) person or organization that creates a software product or package (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.38) Note: This entity might or might not own the rights to sell or distribute the software.

software design. (1) use of scientific principles, technical information, and imagination in the definition of a software system to perform pre-specified functions with maximum economy and efficiency (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

software design audit. (1) review of a software product to determine compliance with requirements, standards, and contractual documents (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software design concept. (1) fundamental idea (such as information hiding) that can be applied to designing a system (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software design description (SDD). (1) representation of software created to facilitate analysis, planning, implementation, and decision-making (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.28) Note: The software design description is used as a medium for communicating software design information and can be thought of as a blueprint or model of the system.
**software design notation.** (1) means of describing a software design (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It can be diagrammatic, symbolic, or textual. Syn: software design representation

**software design verification.** (1) evaluation of a design to determine correctness with respect to stated requirements, conformance to design standards, system efficiency, and other criteria (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**software developer.** (1) person who creates software (ISO/IEC/IEEE 24765e:2015) Note: Often a software developer works with other developers for a software manufacturer to create commercial applications. A software developer can also often work as an in-house developer of software for use by the software developer's own organization.

**software development cycle.** (1) period of time that begins with the decision to develop a software product and ends when the software is delivered (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This cycle typically includes a requirements phase, design phase, implementation phase, test phase, and sometimes, installation and checkout phase. The phases listed above can overlap or be performed iteratively, depending upon the software development approach used. This term is sometimes used to mean a longer period of time, either the period that ends when the software is no longer being enhanced by the developer, or the entire software life cycle. See also: software life cycle

**software development environment.** (1) facilities, hardware, software, procedures, and documentation needed to perform software development (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.14) Note: Plans for software development environments can include where the specified environment is to be constructed, when sites provide different environments or facilities. For example, different testing environments can be requested to be constructed at the acquirer's site and the supplier's site. See also: enabling system

**software development file (SDF).** (1) collection of material pertinent to the development of a given software unit or set of related units (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Contents typically include the requirements, design, technical reports, code listings, test plans, test results, problem reports, schedules, and notes for the units. Syn: software development folder, software development notebook, unit development folder

**software development library.** (1) software library containing computer readable and human readable information relevant to a software development effort (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: project library, program support library See also: master library, production library, software repository, system library

**software development plan (SDP).** (1) information item that describes the technical approach to be followed for a software development effort (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.15) Note: The software development plan presents how the organization or project plans to conduct development activities. A distinction is being made between the technical and management approaches. See also: project management plan

**software development process.** (1) process by which user needs are translated into a software product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes, installing and checking out the software for operational use. These activities can overlap
or be performed iteratively.

**software diversity.** (1) software development technique in which two or more functionally identical variants of a program are developed from the same specification by different programmers or programming teams with the intent of providing error detection, increased reliability, additional documentation, or reduced probability that programming or compiler errors will influence the end results (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**software element.** (1) system element that is software (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.51) (2) identifiable part of a software product. (ISO/IEC/IEEE 90003:2018 Software engineering -- Guidelines for the application of ISO 9001:2015 to computer software, 3.8) See also: system element, software/system element


**software engineering environment (SEE).** (1) environment that provides automated system context services and software-specific services for the engineering of software systems and related domains, such as project management and process management (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.7) (2) hardware, software, and firmware used to perform a software engineering effort (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: It includes the platform, system software, utilities, and CASE tools installed. Syn: infrastructure

**software entitlement.** (1) software license use rights as defined through agreements between a software licensor and a software consumer (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.39) Note: Effective use rights take into account any contracts and all applicable licenses, including full licenses, upgrade licenses, and maintenance agreements.

**software entitlement reconciliation.** (1) process of comparing software entitlements owned with those required (granted and deployed), usually to determine compliance with software license agreements (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.30)

**software feature.** (1) software characteristic specified or implied by requirements documentation (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**software function.** (1) implementation of an algorithm in the software with which the end user or the software can perform part or all of a work task (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.14) Note: a function does not need to be callable by the end user (e.g., automatic backup or data saving).

**software hazard.** (1) software condition that is a prerequisite to an accident (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.5) See also: system hazard
**software identification tag.** (1) Information structure containing identification information about a software configuration item, which can be authoritative if provided by a software creator (*ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.40*). (2) Set of structured data elements containing authoritative identification information about a software configuration item (*ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.31*) Syn: SWID tag, SWID

**software item.** (1) Aggregation of software, such as a computer program or database, that satisfies an end use function and is designated for specification, qualification testing, interfacing, configuration management, or other purposes (*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*). (2) Source code, object code, control code, control data, or a collection of these items (*ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.53*). (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.47) (3) Identifiable part of a software product (*ISO/IEC/IEEE 90003:2018 Systems and software engineering--Life cycle management, 2.1*).

**software library.** (1) Controlled collection of software and related documentation designed to aid in software development, use, or maintenance (*ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary*) Syn: program library See also: repository

**software license.** (1) Legal rights to use software in accordance with terms and conditions specified by the software licensor (*ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.41*) Note: Using a software product can include accessing, copying, distributing, installing, and executing the software product, depending on the license's terms and conditions

**software licensee.** (1) Person or organization granted a license to use a specific software product (*ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.33*)

**software licensor.** (1) Person or organization who owns or holds the rights to issue a software license for a specific software package (*ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.43*)

**software life cycle (SLC).** (1) Project-specific sequence of activities that is created by mapping the activities of a standard onto a selected software life cycle model (SLCM) (*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*). (2) Software system or software product cycle initiated by a user need or a perceived customer need and terminated by discontinued use of the product or when the software is no longer available for use (*ISO/IEC/IEEE 24765e:2015*)

**software maintenance.** (1) Totality of activities required to provide cost-effective support to a software system (*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.1*) (2) Entitlement of additional rights (such as additional functionality, upgrade or support) for a previously granted software entitlement (*ISO/IEC 19770-3:2016*)
Information technology--IT asset management--Part 3: Entitlement schema, 3.1.35) Note: Pre-delivery activities include planning for post-delivery operations, supportability, and logistics determination. Post-delivery activities include software modification, training, and operating a help desk.

**software monitor.** (1) software tool that executes concurrently with another program and provides detailed information about the execution of the other program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: hardware monitor, monitor

**software non-functional assessment process (SNAP) category.** (1) group of components, processes or activities that are used in order to meet the non-functional requirement (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: Categories classify the non-functional requirements (NFR), and are generic enough to allow for future technologies. Categories are divided into subcategories, so that the SNAP category groups the sub-categories based on the same level of operations or similar type of sizing activities. Syn: SNAP category

**Software Non-functional Assessment Process (SNAP) counting unit (SCU).** (1) component or activity, in which non-functional complexity and size are measured (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: The SCU is identified according to the nature of each sub-category. It may contain both functional and non-functional characteristics; therefore, sizing an SCU is performed for its functional sizing, using function point analysis (FPA), and for its non-functional sizing, using SNAP. Syn: SNAP counting unit

**Software Non-functional Assessment Process (SNAP) point.** (1) unit of measurement to express the size of a non-functional requirement (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: The non-functional size of the software non-functional assessment process (SNAP) counting units (SCUs) is identified in a sub-category. After identifying all the SCUs that are provided to meet the non-functional requirements (NFR), the SNAP points (SPs) of all SCUs are added together to obtain the calculated SPs. Syn: SNAP point, software non-functional assessment process point

**Software Non-functional Assessment Process (SNAP) sub-category.** (1) component, a process or an activity executed within the project, to meet a non-functional requirement (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1) Note: A non-functional requirement may consume more than one sub-category. Syn: SNAP sub-category

**software package.** (1) complete and documented set of software supplied for a specific application or function (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.44) Note: the set of files associated with a specific set of business functionalities that can be installed on a computing device and has a set of specific licensing requirements See also: software product

**software packager.** (1) entity that packages or bundles software created by others (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.45)

**software piracy.** (1) illegal use or copying of software products (ISO/IEC 2382:2015 Information technology --Vocabulary)

Software life cycle processes, 3.1.54) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.48) (2) any of the individual items in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (3) complete set of software designed for delivery to a software consumer or end-user which can include computer programs, procedures and associated documentation and data (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.46) (4) set of computer programs, procedures, database- and other data structure descriptions and associated documentation (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.33) Note: A software product can be designated for delivery, an integral part of another product, or used in development. Software products vary from large customized application software for one customer to standard software packages that are sold off the shelf to millions of customers. See also: software package

software product evaluation. (1) technical operation that consists of producing an assessment of one or more characteristics of a software product according to a specified procedure (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.32)

software project life cycle (SPLC). (1) portion of the entire software life cycle applicable to a specific project (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: It is the sequence of activities created by mapping the activities onto a selected software project life cycle model (SPLCM).

software project life cycle model (SPLCM). (1) framework selected by each using organization on which to map the activities of IEEE Std 1074 to produce the software project life cycle (SPLC) (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software project life cycle process (SPLCP). (1) project-specific description of the process developed by adding the organizational process assets (OPAs) to the software project life cycle (SPLC) and the OPAs (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software quality. (1) capability of a software product to satisfy stated and implied needs when used under specified conditions (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.33) (2) degree to which a software product satisfies stated and implied needs when used under specified conditions (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models, 4.3.13) (3) degree to which a software product meets established requirements (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: Quality depends upon the degree to which the established requirements accurately represent stakeholder needs, wants, and expectations. In SQuaRE standards software quality has the same meaning as software product quality.

software quality assurance (SQA). (1) a set of activities that assess adherence to, and the adequacy of the software processes used to develop and modify software products. SQA also determines the degree to which the desired results from software quality control are being obtained. (Software Extension to the PMBOK(R) Guide Fifth Edition) (2) set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2) Note: A key attribute
of SQA is the objectivity of the SQA function with respect to the project. The SQA function can also be organizationally independent of the project; that is, free from technical, managerial, and financial pressures from the project.

**software quality characteristic.** (1) category of software quality attributes that bears on software quality (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.34) Note: Software quality characteristics can be refined into multiple levels of sub-characteristics and finally into software quality attributes.

**software quality control (SQC).** (1) a set of activities that measure, evaluate and report on the quality of software project artifacts throughout the project life cycle (Software Extension to the PMBOK(R) Guide Fifth Edition)

**software quality evaluation.** (1) systematic examination of the extent to which a software product is capable of satisfying stated and implied needs (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.35)

**software quality management.** (1) coordinated activities to direct and control an organization with regard to software quality (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2)

**software quality requirement.** (1) requirement that a software quality attribute be present in software (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.3.14)

**software release management.** (1) management of the activities surrounding the release of one or more versions of software to one or more customers, including identifying, packaging, and delivering the elements of a product (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) See also: software configuration management, version

**software reliability.** (1) probability that software will not cause the failure of a system for a specified time under specified conditions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered.

**software reliability management.** (1) process of optimizing the reliability of software through a program that emphasizes software error prevention, fault detection and removal, and the use of measurements to maximize reliability in light of project constraints such as resources (cost), schedule, and performance (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**software repository.** (1) software library providing permanent, archival storage for software and related documentation (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) See also: production library, software development library, system library

**software requirement.** (1) software capability needed by a user to solve a problem or to achieve an objective (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) software capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**software requirements analysis.** (1) process of studying user needs to arrive at a definition of system, hardware,
software requirements engineering. (1) science and discipline concerned with analyzing and documenting software requirements. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) software requirements elicitation, analysis, specification, verification, and management. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: It involves transforming system requirements into a description of software requirements, performance parameters, and a software configuration using an iterative process of definition, analysis, trade-off studies, and prototyping. See also: requirements engineering

software requirements management. (1) process of planning and controlling the identification, allocation, and flow-down of requirements from the system level to the module or part level, including interfaces, verification, modifications, and status monitoring. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: requirements management

software requirements phase. (1) software development life-cycle phase during which the requirements for a software product, such as functional and performance capabilities, are defined, documented, and reviewed. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

software requirements review (SRR). (1) review of the requirements specified for one or more software configuration items to evaluate their responsiveness to and interpretation of the system requirements and to determine whether they form a satisfactory basis for proceeding into preliminary design of the configuration items. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) review as in (1) for any software component. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This review is called software specification review by the US Department of Defense. See also: system requirements review

software requirements specification (SRS). (1) documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces. (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.29)

software requirements verification. (1) process of ensuring that the software requirements specification complies with the system requirements, conforms to document standards of the requirements phase, and is an adequate basis for the architectural (preliminary) design phase. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: requirements verification, requirements validation

software safety. (1) freedom from software hazards. (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.6) (2) ability of software to be free from unacceptable risk. (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.21) Note: ability of software to resist failure and malfunctions that can lead to death or serious injury to people, loss or severe damage to property, or severe environmental harm. See also: system safety


software security. (1) ability of software to protect its assets from a malicious attacker. (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.22) Note: Software security applies to software assets and is decomposed into the following set of properties: confidentiality, integrity,
availability, authentication, authorization, and non-repudiation.

**software system.** (1) system for which software is of primary importance to the stakeholders *(ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.55)* See also: software-intensive system

**software system element.** (1) member of a set of elements that constitute a software system *(ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.56)* Note: A software system element can include one or more software units, software elements, hardware units, hardware elements, services, and other system elements and systems. A software system element can be viewed as a system element.

**software test environment (STE).** (1) facilities, hardware, software, firmware, procedures, and documentation needed to perform qualification or other testing of software *(ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.15)* Note: Elements include simulators, code analyzers, test case generators, path analyzers, and elements used in the software engineering environment

**software test incident.** (1) event occurring during the execution of a software test that requires investigation *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)*

**software testing.** (1) activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component *(IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2)* *(2) dynamic verification that a program provides expected behaviors on a finite set of test cases, suitably selected from the usually infinite executions domain *(ISO/IEC TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK), 4)*

**software testing environment.** (1) facilities, hardware, software, firmware, procedures, and documentation needed to perform testing of software *(IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes)*

**software tool.** (1) computer program used in the development, testing, analysis, or maintenance of a program or its documentation *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)*

**software transition.** (1) controlled and coordinated sequence of actions wherein software development passes from the organization performing initial software development to the organization performing software maintenance *(IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.10)*

**software unit.** (1) atomic-level software component of the software architecture that can be subjected to standalone testing *(ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.57) *(ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.49)*

**software usage.** (1) consumption against a software entitlement measured as defined by the terms and conditions of that entitlement *(ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.47)* Note: Depending on the specific terms and conditions, usage can include accessing, copying, distributing, installing and executing software.

**software user documentation.** (1) electronic or printed body of material that provides information to users of software *(ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of software)*
software version ID. (1) explicit and immutable version identifier (name or number) inserted into each configuration item, including each individual release, that can be used to identify the exact version of the configuration item in any instance or repository (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1) Syn: software version identification

software-based system. (1) computer system controlled by software (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

software-intensive system. (1) system for which software is a major technical challenge and is perhaps the major factor that affects system schedule, cost, and risk (IEEE 1062-2015 IEEE Recommended Practice for Software Acquisition, 3.1) Note: In the most general case, a software-intensive system is comprised of hardware, software, people, and manual procedures. See also: software system

software/system element. (1) element that defines and prescribes what a software or system is composed of (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.12) Note: An element can contain sub elements or other software/system elements that are dependent on the top-level element. See also: software element

SOI. (1) system-of-interest (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.2) Syn: SoI

SOIC. (1) small outline integrated circuit (ISO/IEC/IEEE 24765c:2014)

solution domain. (1) environment in which a solution or set of solutions resides (ISO/IEC/IEEE 24765e:2015) See also: problem domain

SOO. (1) statement of objectives (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2)


SoPC. (1) system on a programmable chip (ISO/IEC/IEEE 24765d:2015)

sort. (1) symbol representing the name of a set (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.33)

sorting. (1) activity of sequencing of rows or records in a transactional function (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.48)

SoS. (1) system of systems (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.2)

SoSE. (1) system of systems engineering (ISO/IEC/IEEE 21839:2019 Systems and software engineering-- System of systems (SoS) considerations in life cycle stages of a system, 3.2)

SOTIF. (1) safety of the intended functionality (ISO/IEC TR 29119-11:2020, Software and systems engineering-- Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.2)

source address. (1) address of a device or storage location from which data is to be transferred (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: destination address
source code. (1) computer instructions and data definitions expressed in a form suitable for input to an assembler, compiler, or other translator (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A source program is made up of source code. See also: object code

source code organization. (1) arrangement of source code, including layout of code within a single file and packaging of source code into modules, classes, physical files, and so on (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

source document. (1) standard or document defining an existing industry practice (ISO/IEC 19770-8:2020, Information technology  IT asset management  Part 8: Guidelines for mapping of industry practices to/from the ISO/IEC 19770 family of standards, 3.2)

source language. (1) language in which the input to a machine-aided translation process is represented (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) language of the source from which content is rendered into the target language (ISO/IEC/IEEE 26511:2018 Systems and software engineering—Requirements for managers of information for users of systems, software, and services, 3.1.27) See also: target language

source node. (1) node associated with the start of an arc (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.17)

source program. (1) computer program that must be compiled, assembled, or otherwise translated in order to be executed by a computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: object program

source selection criteria. (1) a set of attributes desired by the buyer which a seller must meet or exceed to be selected for a contract (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

source statements (SS). (1) encoded logic of the software product (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Source statements can be classified by function as executable, data declaration, compiler directive, or comment. They can also be classified as deliverable or nondeliverable.


SP. (1) SNAP (software non-functional assessment process) point (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.2)

spawn action. (1) dividing action, where the enabled chains will not join (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.6)

special cause. (1) source of variation that is not inherent in the system, is not predictable, and is intermittent (ISO/IEC/IEEE 24765c:2014) Syn: assignable cause See also: common cause

special function register (SFR). (1) register used to control or monitor functions in a microprocessor (ISO/IEC/IEEE 24765e:2015)

specialization. (1) specification of targeted document type definitions that share the common output transformations and design rules developed for more general types and domains (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.30)
specialize. (1) change by an instance from being an instance of its current class to being additionally an instance of one (or more) of the subclasses of the current subclass (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.181) Note: A specialized instance acquires a different (lower) lowclass. See also: respecialize; unspecialize

specialty engineering. (1) Analysis of specific features of a system that requires special skills to identify requirements and assess their impact on the system life cycle (INCOSE Systems Engineering Handbook, 4th ed.)

specific symbol. (1) symbol used when the precise nature or form of, for example, the process or data media is known and when it is necessary to depict the actual medium (ISO 5807:1985 Information processing -- Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts, 3.2)

specification. (1) detailed formulation, in document form, which provides a definitive description of a system for the purpose of developing or validating the system (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) information item that identifies, in a complete, precise, and verifiable manner, the requirements, design, behavior, or other expected characteristics of a system, service, or process (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.24) (3) a document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system, component, product, result, or service and, often, the procedures for determining whether these provisions have been satisfied. Examples are: requirement specification, design specification, product specification, and test specification (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (4) concrete representation of a model in some notation (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 7.4)

specification change notice (SCN). (1) a document used in configuration management to propose, transmit, and record changes to a specification (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control, engineering change, notice of revision

specification language. (1) a language, often a machine-processible combination of natural and formal language, used to express the requirements, design, behavior, or other characteristics of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: programming language, query language

specification limits. (1) the area, on either side of the centerline, or mean, of data plotted on a control chart that meets the customer's requirements for a product or service. This area may be greater than or less than the area defined by the control limits. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: control limits

specification tree. (1) diagram that depicts the specifications for a given system and shows their relationships to one another (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

specification-based testing. (1) testing in which the principal test basis is the external inputs and outputs of the test item, commonly based on a specification, rather than its implementation in source code or executable software (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.39) Syn: black-box testing, closed-box testing See also: functional testing

SPI. (1) schedule performance index (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) --

spiral model. (1) model of the software development process in which the constituent activities, typically requirements analysis, preliminary and detailed design, coding, integration, and testing, are performed iteratively until the software is complete (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: waterfall model, incremental development, rapid prototyping

SPLC. (1) software project life cycle (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

SPLCM. (1) software project life cycle model (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

SPLCP. (1) software project life cycle process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

split key. (1) foreign key containing two or more attributes, where at least one of the attributes is a part of the entity’s primary key and at least one of the attributes is not a part of the primary key (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.183) Note: [key style]


sponsor. (1) a person or group who provides resources and support for the project, program, or portfolio and is accountable for enabling success (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

sponsoring organization. (1) the entity responsible for providing the project’s sponsor and a conduit for project funding or other project resources (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

spool. (1) to read input data, or write output data, to auxiliary or main storage for later processing or output, in order to permit input/output devices to operate concurrently with job execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: derived from the acronym SPOOL for Simultaneous Peripheral Output On Line

spooler. (1) program that initiates and controls spooling (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

spreadsheet program. (1) program that displays a table of cells arranged in rows and columns, in which the change of the contents of one cell can cause recomputation of one or more cells based on user-defined relations among the cells (ISO/IEC 2382:2015 Information technology -- Vocabulary)

sprint. (1) short time frame, in which a set of software features is developed, leading to a working product that can be demonstrated to stakeholders (ISO/IEC/IEEE 24765h:2019) Note: In some organizations, a sprint is known as an iteration.


SQAP. (1) software quality assurance plan (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.3)

SQC. (1) software quality control (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.2)


squiggle. (1) short ‘&’; ‘&’-shaped line attached at one end to an arrow label and at the other end to an arrow segment (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.101) Note: A squiggle binds an object type set (arrow label) to an object set (arrow segment).


SRAM. (1) static random access memory (ISO/IEC/IEEE 24765c:2014)

SRD. (1) system requirements document (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)


SSE. (1) system security engineering (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)


SSOP. (1) shrink small outline package (ISO/IEC/IEEE 24765c:2014) See also: SOIC

SSPL. (1) software and systems product line (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 4)

SSR. (1) software specification review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2) See also: software requirements review

SSS. (1) system/subsystem specification (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1)

stability. (1) property that an object has with respect to a given failure mode if it cannot exhibit that failure mode (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.6.4)
**stability schema.** (1) specification of failure modes which an object will not exhibit (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 16.2.1.1)

**stabilization phase.** (1) the time interval of the measurement procedure when the RTE starts submitting tasks until the SUT reaches a stable state of operation (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.16)

**stable branch.** (1) branch where stability-disrupting changes are discouraged (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: the branch used for releasing the product's stable production version.

**stable process.** (1) process from which all special causes of process variation have been removed and prevented from recurring, so that only common causes of process variation of the process remain (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**stack pointer.** (1) register that stores the address at the top of a stack (the address of the most recent program request) (ISO/IEC/IEEE 24765d:2015)

**staff-hour.** (1) hour of effort expended by a member of the staff (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


**staged representation.** (1) structure wherein attaining the goals of a set of process areas establishes a maturity level; each level builds a foundation for subsequent levels (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**stakeholder.** (1) individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.59) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.44) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.51) (2) an individual, group, or organization who may affect, be affected by, or perceive itself to be affected by a decision, activity, or outcome of a project, program, or portfolio (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) [system] individual, team, organization, or classes thereof, having an interest in a system (ISO/IEC/IEEE 42010:2011 Systems and software engineering--Architecture description, 3.10) (4) person or organization that can affect, be affected by, or perceive itself to be affected by a decision or activity (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.52) (5) role, position, individual or organization having a right, share, claim or other interest in an architecture entity or its architecture that reflects their needs and expectations (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.20) (6) anyone or any organization that is meaningfully or potentially meaningfully
be impacted by
the autonomous/intelligent system (A/IS) or meaningfully or potentially meaningfully impacts the A/IS (*IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1*)

Note: Some stakeholders can have interests that oppose each other or oppose the system. Syn: interested party

**stakeholder analysis.** (1) a technique of systematically gathering and analyzing quantitative and qualitative information to determine whose interests should be taken into account throughout the project (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)

**stakeholder engagement assessment matrix.** (1) a matrix that compares current and desired stakeholder engagement levels (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)

**stakeholder engagement plan.** (1) a component of the project management plan that identifies the strategies and actions required to promote productive involvement of stakeholders in project or program decision making and execution (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)

**stakeholder equity.** (1) degree of the share or claim a stakeholder has in the system of interest or a portion of the system of interest (*ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.9*)

**stakeholder register.** (1) a project document including the identification, assessment and classification of project stakeholders (*A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition*)

**stakeholder requirements specification.** (1) structured collection of the requirements [characteristics, context, concepts, constraints and priorities] of the stakeholder and the relationship to the external environment (*ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.1.29*)

**stakeholder satisfaction.** (1) degree to which stakeholder needs are satisfied when a product or system is used in a specified context of use (*ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.20*)

**stand-alone.** (1) pertaining to hardware or software that is capable of performing its function without being connected to other components (*ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary*)

**stand-up meeting.** (1) brief daily project status or planning meeting used in agile development methodologies (*ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.14*)

**standard.** (1) document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context (*ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.33*)

(2) document established by an authority, custom, or general consent as a model or example (*ISO/IEC/IEEE 24765h:2019*)

See also: guide, guideline

**standard process.** (1) set of definitions of the processes used to guide processes in an organization (*ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.1.8*)

Note: These process definitions cover the fundamental process elements (and their relationships to each other) that must be incorporated into the defined processes that are implemented in projects across the organization. A standard process establishes
consistent activities across the organization and is desirable for long-term stability and improvement. The organization's set of standard processes describes the fundamental process elements that will be part of the projects' defined processes. It also describes the relationships (for example, ordering and interfaces) between these process elements.

**Standard Technical English (STE).** (1) controlled language that includes a set of writing rules and a basic dictionary for writing technical documentation (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.28) Note: The STE specification is maintained by the ASD Simplified Technical English Maintenance Group (STEMG).

**standardized profile.** (1) internationally agreed-to, harmonized standard which describes one or more profiles (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.12)

**standby mode.** (1) operating mode which saves power when a microcontroller unit is not in active use (ISO/IEC/IEEE 24765d:2015) Syn: sleep mode

**standby redundancy.** (1) in fault tolerance, the use of redundant elements that are left inoperative until a failure occurs in a primary element (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: active redundancy

**start date.** (1) a point in time associated with a schedule activity's start, usually qualified by one of the following: actual, planned, estimated, scheduled, early, late, target, baseline, or current. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**start-to-finish (SF).** (1) a logical relationship in which a successor activity cannot finish until a predecessor activity has started. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: logical relationship

**start-to-start (SS).** (1) a logical relationship in which a successor activity cannot start until a predecessor activity has started (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: logical relationship

**starting address.** (1) address of the first instruction of a computer program in main storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This address can be the same as the program's origin, depending upon whether there are data preceding the first instruction. See also: origin, assembled origin, loaded origin

**state.** (1) at a given instant in time, the condition of an object that determines the set of all sequences of actions (or traces) in which the object can participate (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 8.8) (2) condition that characterizes the behavior of a function, subfunction or element at a point in time (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 3.1.30) (3) unique value that represents the stage of progress of software in its execution (ISO/IEC 11411:1995 Information technology -- Representation for human communication of state transition of software, 2.1) Note: A state is a shorthand representation for the unit's interaction occurrence history. Ascribing "state" to a unit implies that it has a capability to modify future behaviors as a result of past interactions with its environment.

**state class.** (1) class that represents a set of real or abstract objects that have common knowledge or behavior (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97)
A state class represents instances with changeable state. The constituent instances of a state class can come and go and can change state over time; that is, their property values can change.

**state data.** (1) data that defines an internal state of the test unit and is used to establish that state or compare with existing states *(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)*

**state diagram.** (1) diagram that depicts the states that a system or component can assume, and shows the events or circumstances that cause or result from a change from one state to another *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**state invariant condition.** (1) statement of constraints or relations that can be used to distinguish a subset of particular property states that all satisfy the condition *(ISO/IEC/IEEE 24765i:2020)*

**state name.** (1) unique identifier of the state of software execution *(ISO/IEC 11411:1995 Information technology -- Representation for human communication of state transition of software, 2.1)*

**statement.** (1) in a programming language, a meaningful expression that defines data, specifies program actions, or directs the assembler or compiler *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: assignment statement, control statement, declaration

**statement coverage.** (1) percentage of the set of all executable statements of a test item that are covered by a test set *(ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.40)*

**statement of work (SOW).** (1) a narrative description of products, services, or results to be delivered by the project *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)* *(2)* means to describe and specify the tasks to be performed under the contract *(ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.65)* *(3)* document used by the acquirer that includes the needs and expectations, the scope, objectives and deliverables *(ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.11)*

**statement testing.** (1) testing designed to execute each statement of a computer program *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* test design technique in which test cases are constructed to force execution of individual statements in a test item *(ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.41)* See also: branch testing, path testing

**static.** (1) pertaining to an event or process that occurs without computer program execution *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: dynamic

**static analysis.** (1) process of evaluating a system or component based on its form, structure, content, or documentation *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: dynamic analysis, inspection, walk-through

**static binding.** (1) binding performed prior to the execution of a computer program and not subject to change during program execution *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: dynamic binding

**static breakpoint.** (1) breakpoint that can be set at compile time, such as entry into a given routine *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: dynamic breakpoint, code breakpoint, data
static dump. (1) dump that is produced before or after the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dynamic dump, change dump, memory dump, postmortem dump, selective dump, snapshot dump

static error. (1) error that is independent of the time-varying nature of an input (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: dynamic error

static model. (1) model that describes an interrelated set of classes (and/or subject domains) along with their relationships and responsibilities (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.185) See also: dynamic model

static product. (1) non-executable system or software product for reviewing (ISO/IEC 25041: 2012 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation guide for developers, acquirers and independent evaluators, 4.12)

static program analysis. (1) sub-field of formal methods concerned with analyzing the properties of software code without executing this code in the target (binary) format (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.24)

static random access memory (SRAM). (1) random access memory without refresh process, which keeps data as long as it is powered on (ISO/IEC/IEEE 24765c:2014) Note: based on a circuit architecture with double stable states

static schema. (1) specification of the state of one or more information objects, at some point in time, subject to the constraints of any invariant schemata (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 6.1.2) Note: Thus, a static schema is the specification of the types of one or more information objects at some particular point in time. These types are subtypes of the types specified in the invariant schema.

static testing. (1) testing in which a test item is examined against a set of quality or other criteria without code being executed (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.42) See also: inspection

statistical process control. (1) statistically based analysis of a process and measures of process performance, which identify common and special causes of variation in process performance and maintain process performance within limits (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

statistical sampling. (1) choosing part of a population of interest for inspection (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

statistically managed process. (1) process that is managed by a statistically based technique in which processes are analyzed, special causes of process variation are identified, and performance is contained within well-defined limits (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

status code. (1) code used to indicate the results of a computer program operation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: condition code

of information for users of systems, software, and services, 3.2)

**step.** (1) one element (numbered list item) in a procedure that tells a user to perform an action (or actions) *(ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.47)

(2) one element in a procedure that tells the target audience to perform an action *(IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.37) *(3) abstraction of an action, used in a process, which can leave unspecified some or all of the objects that participate in that action *(ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.6) *(4) element of a procedure containing one or more actions that enables a user to perform a task *(ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.36) Note: A step contains one or more actions. A step is presented as a numbered list item. Responses by the software are not considered to be steps.

**step-by-step instructions.** (1) procedure consisting of steps to be followed for the safe, effective, and efficient use of a supported product *(IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.38)

**stepwise refinement.** (1) software development technique in which data and processing steps are defined broadly at first and then further defined with increasing detail *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, structured design, transaction analysis, transform analysis

**stimulus.** (1) whatever causes a unit to exhibit an occurrence of a behavior pattern in the unit's repertoire; something causing or regarded as causing a response *(ISO/IEC/IEEE 24765i:2020) Note: A stimulus can be the occurrence of an externally controlled interaction affecting the unit (external stimulus), or it can be the occurrence of an internally controlled event (internal stimulus).


**stock keeping unit (sku).** (1) identification, usually alphanumeric, of a particular product that allows it to be tracked for inventory and software entitlement purposes *(ISO/IEC 19770-5:2015 Information technology -- IT asset management--Overview and vocabulary, 3.48) Note: typically associated with unique products for sales purposes, such as software entitlements. It can correspond uniquely to specific software products, or represent packages of software, with specific terms and conditions, such as whether it relates to a full product, upgrade product, or maintenance on an existing product.

**stop.** (1) to terminate the execution of a computer program *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: halt *(1) See also: pause

**storage.** (1) functional unit into which data can be placed, in which data can be retained, and from which data can be retrieved *(ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: storage device

**storage allocation.** (1) element of computer resource allocation, consisting of assigning storage areas to specific jobs and performing related procedures, such as transfer of data between main and auxiliary storage, to support the assignments made *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: buffer,
contiguous allocation, cyclic search, memory compaction, overlay, paging, virtual storage

**storage capacity.** (1) maximum number of items that can be held in a given storage device; usually measured in words or bytes *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**storage efficiency.** (1) degree to which a system or component performs its designated functions with minimum consumption of available storage *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**store.** (1) to place or retain data in a storage device *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(2)* to copy computer instructions or data from a register to internal storage or from internal storage to external storage *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**story point.** (1) the relative measure of the effort needed to develop a user story, compared with what is considered a typical user story by the project team *(Software Extension to the PMBOK(R) Guide Fifth Edition)*

**STP.** (1) software test plan *(IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)*

**straight-line code.** (1) sequence of computer instructions in which there are no loops *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

**straight-line coding.** (1) programming technique in which loops are avoided by stating explicitly and in full all of the instructions that would be involved in the execution of each loop *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(See also: unwind)*

**strategic asset management plan (SAMP).** (1) documented information that specifies how organizational objectives are to be converted into asset management objectives, the approach for developing asset management plans, and the role of the asset management system in supporting achievement of the asset management objectives *(ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.53)*

**strategic IT asset management plan.** (1) documented information that specifies how organizational objectives are to be converted into IT asset management objectives, the approach for developing IT asset management plans, and the role of the IT asset management system in supporting achievement of the IT asset management objectives *(ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.54)* *(Syn: strategic information technology asset management plan)*

**strategy.** (1) organization's overall plan of development, describing the effective use of resources in support of the organization in its future activities *(ISO/IEC/IEEE 24765c:2014)* *(Note: involves setting objectives and proposing initiatives for action)*

**stratified language.** (1) language that cannot be used as its own metalanguage *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* *(See also: unstratified language)*

**stream interface.** (1) interface in which all the interactions are flows *(ISO/IEC 10746-3:2009 Information technology - - Open Distributed Processing -- Reference Model: Architecture, 7.1.8)*

**stream interface signature.** (1) interface signature for a stream interface *(ISO/IEC 10746-3:2009 Information technology - - Open Distributed Processing -- Reference Model: Architecture, 7.1.8)*
A stream interface comprises a finite set of action templates, one for each flow type in the stream interface. Each action template for a flow contains the name of the flow, the information type of the flow, and an indication of causality for the flow (i.e. producer or consumer but not both) with respect to the object which instantiates the template. The phrase "complementary interface signature to X", where X is itself an interface signature describes an interface signature identical to X in all respects except causality, which is opposite to that in X. Many Interface Definition Languages (IDLs) capture only the action templates of a signature and depend upon the context in which the IDL is used to determine the causality that is to be applied.

**strengths, weaknesses, opportunities, and threats (SWOT) analysis.** (1) analysis of strengths, weaknesses, opportunities and threats of an organization, project, or option (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**stress testing.** (1) type of performance efficiency testing conducted to evaluate a test item's behavior under conditions of loading above anticipated or specified capacity requirements, or of resource availability below minimum specified requirements (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.43) See also: boundary value

**strong typing.** (1) feature of some programming languages that requires the type of each data item to be declared, precludes the application of operators to inappropriate data types, and prevents the interaction of data items of incompatible types (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**StRS.** (1) stakeholder requirements specification (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)

**structural testing.** (1) testing that takes into account the internal mechanism of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Types include branch testing, path testing, statement testing. See also: functional testing (1), structure-based testing

**structure chart.** (1) diagram that identifies modules, activities, or other entities in a system or computer program and shows how larger or more general entities break down into smaller, more specific entities (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The result is not necessarily the same as that shown in a call graph. Syn: hierarchy chart, program structure chart See also: call graph

**structure clash.** (1) in software design, a situation in which a module must deal with two or more data sets that have incompatible data structures  (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data structure-centered design, order clash

**structure-based testing.** (1) dynamic testing in which the tests are derived from an examination of the structure of the test item (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.46) Note: Structure-based testing is not restricted to use at component level and can be used at all levels, e.g. menu item coverage as part of a system test. Syn: glass-box testing, white-box testing. See also: structural testing

**structured authoring.** (1) development of content elements including metadata in specified templates (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.31) Note: In structured authoring, content elements are labeled according to the nature of the content they contain. Structured authoring also permits quasi-semantic labeling, such as Heading1 or
Structured design. (1) disciplined approach to software design that adheres to specified rules based on principles such as modularity, top-down design, and stepwise refinement of data, system structures, and processing steps (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) result of applying the approach in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping.

Structured program. (1) computer program constructed of a basic set of control structures, each having one entry and one exit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The set of control structures typically includes: sequence of two or more instructions, conditional selection of one of two or more sequences of instructions, and repetition of a sequence of instructions. See also: structured design

Structured programming. (1) software development technique that includes structured design and results in the development of structured programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

Structured programming language. (1) programming language that provides the structured program constructs, namely, single-entry-single-exit sequences, branches, and loops, and facilitates the development of structured programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: block-structured language

Structured walkthrough. (1) systematic examination of the requirements, design, or implementation of a system, or any part of it, by qualified personnel (ISO/IEC 2382:2015 Information technology -- Vocabulary)

Stub. (1) skeletal or special-purpose implementation of a software module, used to develop or test a module that calls or is otherwise dependent on it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) computer program statement substituting for the body of a software module that is or will be defined elsewhere (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) engineering object in a channel, which interprets the interactions conveyed by the channel, and performs any necessary transformation or monitoring based on this interpretation (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 8.1.9) (4) scaffolding code written for the purpose of exercising higher-level code before the lower-level routines that will ultimately be used are available (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


Sub-path. (1) path that is part of a larger path (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.26)

Subactivity. (1) subgraph of an activity which is itself an activity and which satisfies the following condition: for any pair of fork-join actions in the parent activity, if one of these actions is included in the subgraph, then both must be included in the subgraph (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.1.8)
subclass. (1) specialization of one or more superclasses (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.186) (2) relation between class A and class B in which the type associated with A is a subtype of the type associated with B (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.12) Note: Each instance of a subclass is an instance of every superclass. A subclass typically specifies additional, different responsibilities to those of its superclasses or overrides superclass responsibilities to provide a different realization. See also: category entity, subtype, superclass

subclass cluster. (1) set of one or more generalization structures in which the subclasses share the same superclass and in which an instance of the superclass is an instance of no more than one subclass (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.187) (2) set of one or more mutually exclusive specializations of the same generic entity (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.187) Note: A cluster exists when an instance of the superclass can be an instance of only one of the subclasses in the set, and each instance of a subclass is an instance of the superclass. Syn: category cluster

subclass responsibility. (1) designation that a property of a class must be overridden in its subclasses (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.188) Note: That is, the designation given to a property whose implementation is not specified in this class. A property that is a subclass responsibility is a specification in the superclass of an interface that each of its subclasses must provide. A property that is designated as a subclass responsibility has its realization deferred to the subclass(es) of the class.

subdomain. (1) domain which is a subset of a given domain (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 10.4)

subject area. (1) related collection of meta-object instance definitions (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: Subject areas are used to define scoped areas of interest. Subject areas overlap to ensure the integration of the overall metamodel, but a tool need only use those subject areas relevant to the data to be exported or imported.

subject domain. (1) area of interest or expertise (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.189) Note: The responsibilities of a subject domain are an aggregation of the responsibilities of a set of current or potential named classes. A subject domain can also contain other subject domains. A subject domain encapsulates the detail of a view.

subject domain responsibility. (1) generalized concept that the analyst discovers by asking, "In general, what do instances in this subject domain need to be able to do or to know?" (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.190) Note: The classes and subject domains in a subject domain together supply the knowledge, behavior, and rules that make up the subject. These notions are collectively referred to as the subject domain's responsibilities. Subject domain responsibilities are not distinguished as sub-domains or classes during the early stages of analysis.

subject system. (1) computing system (existing or to be created) about which descriptive information is being
developed in a computing system or CASE tool (IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.1) (IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection--Characterization of Interconnections, 3.13)

submit primitive. (1) service primitive for which the protocol object is the initiating object of the corresponding communication (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.12)

subnetwork. (1) subdivision (fragment) of a project schedule network diagram, usually representing a sub-project or a work package (ISO/IEC/IEEE 24765h:2019) Note: often used to illustrate or study some potential or proposed schedule condition, such as changes in preferential schedule logic or project scope

subprogram. (1) separately compilable, executable component of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages. See also: coroutine, main program, routine, subroutine

subproject. (1) portion of the overall project created when a project is subdivided into more manageable parts (ISO/IEC/IEEE 24765h:2019)

subroutine. (1) routine that returns control to the program or subprogram that called it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages. See also: coroutine, closed subroutine, open subroutine

subroutine trace. (1) record of all or selected subroutines or function calls performed during the execution of a computer program and, optionally, the values of parameters passed to and returned by each subroutine or function (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: call trace See also: execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

subscription-based license. (1) license for an entitlement that is for a limited amount of time (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.39) Note: It is not a perpetual license and requires renewal to remain in force. Syn: term-based license, service-based license

substitutability. (1) principle stating that, since each instance of a subclass is an instance of the superclass, an instance of the subclass is acceptable in any context where an instance of the superclass is acceptable (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEFIX97 (IDEFObject), 3.1.191) Note: Any request sent to an instance receives an acceptable response, regardless of whether the receiver is an instance of the subclass or the superclass.

subsystem. (1) secondary or subordinate system within a larger system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

subtype. (1) subset of a data type, obtained by constraining the set of possible values of the data type (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) meta-entity that inherits all of the meta-attributes and meta-relationships of its immediate and indirect supertype meta-entities (ISO/IEC 15474-2:2002 Information technology -- CDIF framework -- Part 2: Modelling and extensibility, 6.2.5) (3) relation of type A to type B when every &lt;X&gt; which satisfies A also satisfies B (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.11) Note: The operations applicable to the subtype are the same as those of the original data type.

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement "Copyright ©, 2021, IEEE. Used by permission." remains with the definition. All other rights are reserved.
See also: category entity, derived type, subclass, supertype

success criteria. (1) set of conditions to be satisfied by a process instance at completion (ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.9) Note: Information items and artefacts produced by the process instance must meet the success criteria. Success criteria are established based on the outcomes of the corresponding life cycle process, requirements of the system element to which the process instance contributes, and requirements and constraints arising from decisions in other process instances.

successful adoption. (1) extent to which the use of CASE tools can measurably meet an organization's uniquely defined adoption goals (ISO/IEC TR 14471:2007 Information technology--Software engineering--Guidelines for the adoption of CASE tools, 2.1.1)

succession plan. (1) process for identifying and developing current employees with the potential to fill key positions in the organization (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.30)

suitability. (1) degree to which an IT service meets stated and implied needs when used in a specified context of use (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.1)

summative evaluation. (1) evaluation designed to present conclusions about the merit or worth of the object of evaluation (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.21)

sunk cost. (1) cost that is irrecoverable by future actions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Sunk costs have a psychological impact, but are irrelevant in business decisions.

SUP. (1) supplier (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.36)

superclass. (1) class whose instances are specialized into one or more subclasses (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.193) (2) relation between class B and class A, when the type associated with A is a subtype of the type associated with B (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.12) See also: generic entity, partial cluster, total cluster, subclass, supertype

supercomputer. (1) class of computers that have the highest processing speeds available at a given time (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: often used for solving scientific and engineering problems

supertype. (1) relation between type B and type A, in which every X that satisfies A also satisfies B (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.11) See also: generic entity, subtype, supertype

supervised learning. (1) task of learning a function that maps an input to an output based on labelled example input-output pairs (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.74) See also: unsupervised learning

supervisor state. (1) in the operation of a computer system, a state in which the supervisory program is executing
Note: This state usually has higher priority than, and precludes the execution of, application programs. Syn: executive state, privileged state See also: problem state

supervisory program. (1) computer program, usually part of an operating system, that controls the execution of other computer programs and regulates the flow of work in a computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Syn: control program, executive, executive program, supervisor See also: supervisor state, kernel

supplemental Ent. (1) Ent which has an &lt;entType&gt; of Supplemental (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.40) Note: Supplemental Ents provide extended information about a primary Ent and are linked to primary Ents by the &lt;linkedToPrimaryEntId&gt; attribute. Syn: supplemental entitlement schema

supplemental information structure. (1) Information structure that has a subsidiary relationship to another information structure and extends the information in that information structure (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.49) Syn: supplemental info struct

supplementary run. (1) time interval of the measurement procedure from the time the measurement results fulfill the required statistical significance to the time when all tasks, which were submitted during the rating interval, are completed (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.17)

supplier. (1) organization or individual that enters into an agreement with the acquirer for the supply of a product or service (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.60) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.45) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.52) (2) individual or organization that enters into a contract with the acquirer for the supply of a system, software product or software service under the terms of the contract (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.37) (ISO/IEC 25040:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Evaluation process, 4.63) (3) organization or part of an organization or individual that enters into an agreement with the application management organization for the supply of a product, service, materials, or human capacity (ISO/IEC 16350-2015 Information technology--Systems and software engineering--Application management, 4.34) (4) individual or organization that provides products (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.39) Note: The acquirer and the supplier sometimes are part of the same organization. The application management organization can have internal or external suppliers. A supplier can be another application management organization, but also IT infrastructure management organizations or consultants. Syn: agency, contractor, distributor, producer, retailer, seller, SUP, vendor

support. (1) set of activities necessary to ensure that an operational system or component fulfills its original requirements and any subsequent modifications to those requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: software life cycle, system life cycle

support activity group. (1) activity group that is necessary to assure the successful completion of a project, but
consists of supporting activities rather than activities directly oriented to the development effort (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**support manual.** (1) document that provides the information necessary to service and maintain an operational system or component throughout its life cycle (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) 
**Note:** Typically described are the hardware and software that make up the system or component and procedures for servicing, repairing, or reprogramming it. See also: diagnostic manual, installation manual, maintenance manual, operator manual, programmer manual, user manual

**support software.** (1) software that aids in the development or maintenance of other software (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) software or a program that aids in the development, maintenance, or use of other software or provides general application-independent capability (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: application software, system software

**support staff-hour.** (1) hour of effort expended by a member of the staff who does not directly define or create the software product, but acts to assist those who do (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**supported product.** (1) product to which the information for use relates (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.40)

**surveillance.** (1) systematic iteration of conformity assessment activities as a basis for maintaining the validity of the statement of conformity (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)-Part 2-1: Framework and taxonomy, 4.57)

**survivability.** (1) degree to which a product or system continues to fulfill its mission by providing essential services in a timely manner in spite of the presence of attacks (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.6) See also: recoverability

**suspension criteria.** (1) criteria used to (temporarily) stop all or a portion of the testing activities (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.46)

**sustainment.** (1) activities performed to ensure that a product or service remains operational (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: maintenance

**SV.** (1) schedule variance (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**SVD.** (1) software version description (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**SVG.** (1) scalable vector graphics (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)

**SVM.** (1) support vector machine (ISO/IEC/IEEE 24765j:2021)

**SVR.** (1) system verification review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**SW.** (1) software (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)

**swap.** (1) exchange of the contents of two storage areas, usually an area of main storage with an area of auxiliary storage.
storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to perform an exchange as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: roll in, roll out


SWID. (1) software identification (tag) (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.40)


symbol. (1) graphic representation of a concept that has meaning in a specific context (ISO/IEC 2382:2015 Information technology -- Vocabulary)

d symbol table. (1) table that presents program symbols and their corresponding addresses, values, and other attributes (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

symbolic address. (1) address expressed as a name or label that must be translated to the absolute address of the device or storage location to be accessed (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: absolute address

symbolic execution. (1) software analysis technique in which program execution is simulated using symbols, such as variable names, rather than actual values for input data, and program outputs are expressed as logical or mathematical expressions involving these symbols (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

symbolic language. (1) programming language that expresses operations and addresses in symbols convenient to humans rather than in machine language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: machine language

symbolic trace. (1) record of the source statements and branch outcomes that are encountered when a computer program is executed using symbolic, rather than actual, values for input data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: execution trace, retrospective trace, subroutine trace, variable trace

symmetric net. (1) high-level Petri net where types are finite and only built-in operations are allowed (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.34)

symmetric net. (1) high-level Petri net where types are finite and only built-in operations are allowed (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.34)

synchronize. (1) to pull the changes made in a parent branch into its (evolving) child (for example, feature) branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to update a view with the current version of the files in its corresponding branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

synchronous. (1) pertaining to two or more processes that depend upon the occurrence of specific events such as common timing (ISO/IEC 2382:2015 Information technology -- Vocabulary)

synchronous dynamic random access memory (SDRAM). (1) DRAM with memory access driven by a clock that is synchronized with the processor's memory bus clock (ISO/IEC/IEEE 24765c:2014) Note: SDRAM can access...
data fractions in different memory banks simultaneously.

**synchronous graphics random access memory (SGRAM).** (1) SDRAM designed for the graphics card of a computer (ISO/IEC/IEEE 24765c:2014)

**synchronous message communication.** (1) form of communication in which a producer task sends a message to a consumer task and waits for acknowledgment (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: tightly coupled message communication

**synchronous message communication with reply.** (1) form of communication in which a producer (or client) task sends a message to a consumer (or server) task and waits for a reply (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: tightly coupled message communication with reply

**synchronous message communication without reply.** (1) a form of communication in which a producer task sends a message to a consumer task and waits for the consumer to accept the message (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: tightly coupled message communication without reply

**synchronous request.** (1) request where the client pauses to wait for completion of the request (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.17)

**syntactic error.** (1) violation of the structural or grammatical rules defined for a language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: syntax error See also: semantic error

**syntax.** (1) structural or grammatical rules that define how the symbols in a language are to be combined to form words, phrases, expressions, and other allowable constructs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) structural components or features of a language and rules that define the ways in which the language constructs can be assembled together to form sentences (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.195) (3) definition of the format of information in a CDIF transfer (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) See also: semantics

**SYNTAX.1.** (1) primary syntax defined within the CDIF family of standards (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The CDIF family of standards supports multiple transfer formats, each composed of a syntax and an encoding.

**SyRS.** (1) System Requirement Specification (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2) See also: SRS

TR 19759:2016 Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK), 1.1.6) set of interrelated elements considered in a defined context as a whole and separated from their environment (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.41) integrated set of elements, subsystems, or assemblies that accomplish a defined objective (INCOSE Systems Engineering Handbook, 4th ed.) Note: A system is sometimes considered as a product or as the services it provides. In practice, the interpretation of its meaning is frequently clarified by the use of an associative noun, e.g., aircraft system. Alternatively, the word ‘system’ can be replaced by a context-dependent synonym, e.g., aircraft, though this obscures the system perspective. A complete system includes all of the associated equipment, facilities, material, computer programs, firmware, technical documentation, services, and personnel required for operations and support to the degree necessary for self-sufficient use in its intended environment.

**system analysis.** (1) systematic investigation of a real or planned system to determine the information requirements and processes of the system and how these relate to each other and to any other system (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: systems analysis

**system breakdown structure (SBS).** (1) system hierarchy, with identified enabling systems, and personnel that is typically used to assign development teams, support technical reviews, and to partition the assigned work and associated resource allocations to each of the tasks necessary to accomplish the technical objectives of the project (ISO/IEC/IEEE 24748-4:2016 Systems and software engineering-Life cycle management-Part 4: Systems engineering planning, 4.12) Note: It also provides the basis for cost tracking and control.

**system description.** (1) documentation that results from system design defining the organization, essential characteristics and the hardware and software requirements of the system (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**system design.** (1) process of defining the hardware and Software architecture, components, modules, interfaces and data for a system to satisfy specified requirements (ISO/IEC 2382:2015 Information technology -- Vocabulary) **system design review (SDR).** (1) review conducted to evaluate the manner in which the requirements for a system have been allocated to configuration items, the system engineering process that produced the allocation, the engineering planning for the next phase of the effort, manufacturing considerations, and the planning for production engineering (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: critical design review, preliminary design review

**system development.** (1) process that usually includes requirements analysis, system design, implementation, documentation and quality assurance (ISO/IEC 2382:2015 Information technology -- Vocabulary)

**system development cycle.** (1) period of time that begins with the decision to develop a system and ends when the system is delivered to its end user (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: This term is sometimes used to mean a longer period of time, either the period that ends when the system is no longer being enhanced, or the entire system life cycle. See also: system life cycle software development cycle

**system documentation.** (1) collection of documents that describe the requirements, capabilities, limitations, design, operation, and maintenance of an information processing system (ISO/IEC 2382:2015 Information technology -- Vocabulary)
system effectiveness analysis. (1) analytical approach used to determine how well a system performs in its intended utilization environment (IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1)


系统实体. (1) in Mk II FPA, a contrivance which 'lumps together' all the non-primary entities of an application (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10)

系统后续. (1) study of the effects of a system after it has reached a stabilized state of operational use (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: post-implementation review, post-development review

系统危险. (1) system condition that is a prerequisite to an accident (IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.8) See also: software hazard

系统集成. (1) progressive assembling of system components into the whole system (ISO/IEC 2382:2015 Information technology -- Vocabulary)

系统集成模块 (SIM). (1) module in a microcontroller unit (MCU) that controls a system functional unit (ISO/IEC/IEEE 24765e:2015)

系统界面任务. (1) task that hides the interface to and communicates with an external system or subsystem (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

系统库. (1) software library containing system-resident software that can be accessed for use or incorporated into other programs by reference (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: production library, software development library, software repository

系统生命周期. (1) the evolution with time of a system-of-interest from conception through to retirement (INCOSE Systems Engineering Handbook, 4th ed.) (2) course of developmental changes through which a system passes from its conception to the termination of its use (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) period that begins when a system is conceived and ends when the system is no longer available for use (ISO/IEC/IEEE 21840:2019 Systems and software engineering--Guidelines for the utilization of ISO/IEC/IEEE 15288 in the context of system of systems (SoS), 3.1.11)

系统维护. (1) modification of a system to correct faults, to improve performance, or to adapt the system to a changed environment or changed requirements (ISO/IEC 2382:2015 Information technology -- Vocabulary)

系统模型. (1) in computer performance evaluation, a representation of a system depicting the relationships between workloads and performance measures in the system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: workload model

系统之系统 (SoS). (1) system-of-interest (SOI) whose elements are themselves systems (ISO/IEC/IEEE
set of systems that integrate or interoperate to provide a unique capability that none of the constituent systems can accomplish on its own (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.64) Note: Typically, SoS handle large-scale interdisciplinary problems with multiple, heterogeneous, distributed systems. Each constituent system is a useful system by itself, having its own management, goals, and resources, but coordinates within the SoS to provide the unique capability of the SoS.

system profile. (1) set of measurements used in computer performance evaluation, describing the proportion of time each of the major resources in a computer system is busy, divided by the time that resource is available (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

system requirements review (SRR). (1) review conducted to evaluate the completeness and adequacy of the requirements defined for a system; to evaluate the system engineering process that produced those requirements; to assess the results of system engineering studies; and to evaluate system engineering plans (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software requirements review

system requirements specification (SyRS). (1) structured collection of information that embodies the requirements of the system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) structured collection of the requirements (functions, performance, design constraints, and attributes) of the system and its operational environments and external interfaces (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1.29) See also: software requirements specification, SRS

system safety. (1) freedom from system hazards (IEEE Std 1228-1994 IEEE Standard for Software Safety Plans, 3.1.9) (2) ability of a system to be free from unacceptable risk (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.25) Note: A systems-based approach to safety involves the application of scientific, technical, and managerial skills to hazard identification, hazard analysis, and elimination, control, or management of hazards throughout the life-cycle of a system, program, project or an activity or a product. See also: software safety

system software. (1) software designed to facilitate the operation and maintenance of a computer system and its associated programs (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) application-independent software that supports the running of application software (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: application software, support software

system specification. (1) documented set of mandatory requirements for a system (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1)

system structure. (1) decomposition of a system of interest into a set of interacting systems and system elements (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.10) Note: The system structure is described in a System Breakdown Structure (SBS).

system support. (1) continued provision of services and material necessary for the use and improvement of an implemented system (ISO/IEC 2382:2015 Information technology -- Vocabulary)

system table. (1) an entity type that cannot be maintained and, consequently, is not counted within the framework of
FPA (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

system testing. (1) testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements (ISO/IEC/IEEE 24765g:2018)

system under test (SUT). (1) parts of the CBSS to be tested (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.18) Note: The SUT consists of hardware, system software, data communication features or application software or a combination of them.


systematic failure. (1) failure related in a deterministic way to a certain cause that can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation, or other relevant factors (ISO/IEC/IEEE 15286-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.10)

systems engineering (SE). (1) interdisciplinary approach governing the total technical and managerial effort required to transform a set of stakeholder needs, expectations, and constraints into a solution, and to support that solution throughout its life (ISO/IEC 15940:2013 Systems and software engineering--Software Engineering Environment Services, 2.6) (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.65) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.49) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.56) Note: System engineering is used when there is a single system-of-interest; systems engineering is for the discipline in general. It includes the definition of technical performance measures; the integration of engineering specialties toward the establishment of an architecture; and the definition of supporting lifecycle processes that balance cost, performance, and schedule objectives.

Systems Engineering Plan (SEP). (1) top level technical plan for managing the systems engineering effort which defines how the technical aspects of the project will be organized, structured, and conducted and how the systems engineering processes will be controlled to provide a product that satisfies stakeholder requirements (ISO/IEC/IEEE 24748-4:2016 Systems and software engineering-Life cycle management-Part 4: Systems engineering planning, 4.14) (2) top-level plan for managing the SE effort which, as such, defines how the project will be organized, structured, and conducted and how the total engineering process will be controlled to provide a product that satisfies stakeholder requirements (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.5) (3) top-level technical planning document for a project which addresses technical management processes established by three principal sources: the project's contract or agreement, applicable organizational processes, and the systems engineering project team, as necessary to successfully accomplish the systems engineering-related tasks of the project.


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
structured information describing how the systems engineering effort, in the form of tailored processes and activities, for one or more life cycle stages, will be managed and conducted in the organization for the actual project (INCOSE Systems Engineering Handbook, 4th ed.) Syn: Systems Engineering Management Plan, SEMP, System Engineering Plan

**systems integration testing.** (1) testing conducted on multiple complete, integrated systems to evaluate their ability to communicate successfully with each other and to meet the overall integrated systems' specified requirements (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**systems of systems engineering (SoSE).** (1) process of planning, analyzing, organizing, developing and integrating the capabilities of a mix of existing and new systems, including inter-system infrastructure, facilities, and overarching processes into a system-of-systems capability that is greater than the sum of the capabilities of the constituent systems (ISO/IEC/IEEE 21841:2019 Systems and software engineering--Taxonomy of systems of systems, 3.1.5) Note: SoSE also includes testing, modification, maintenance and other post-integration activities.


**T-profile.** (1) Transfer profile (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview)

**table.** (1) more concrete representation of an entity (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.3)

**table heading.** (1) symbolic name or other means of referencing a decision table from other documents (ISO 5806:1984 Information processing -- Specification of single-hit decision tables, 3.12) Note: Alternatively, or in addition, a clear description of the table.

**table of contents.** (1) list of the headings in a document in order of appearance, with location indicators (such as page numbers) shown for each heading (ISO/IEC/IEEE 24765a:2011)

**table-driven method.** (1) scheme that lets a program look up information in a table rather than using logic statements (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**tacit knowledge.** (1) undocumented information (Software Extension to the PMBOK(R) Guide Fifth Edition) (2) personal knowledge that can be difficult to articulate and share such as beliefs, experience, and insights (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**tag.** (1) symbolic name assigned to a specific release or a branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) information structure that provides authoritative information about a software asset in order to facilitate its management (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: provides developers and end users with a unique reference to the code base they are working with.
tag slide. (1) to apply the same tag to a changed version of a file to correct a last-minute error found in a release
(ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)
tagid. (1) globally unique value that is globally unique for every SWID tag created (ISO/IEC 19770-2:2015 (corr 2017),
Information technology -- Software asset management -- Part 2: Software identification tag, 3.1.3)
tailored process. (1) process developed by tailoring a standard process (ISO/IEC 33001:2015 Information
technology--Process assessment--Concepts and terminology, 3.1.10)
tailoring. (1) adaptation of a software process by adding, modifying, and deleting process activities that are deemed
inapplicable for the project (Software Extension to the PMBOK(R) Guide Fifth Edition) (2) process by which individual
requirements in specifications, standards, and related documents are evaluated and made applicable to a specific project
by selection, and in some exceptional cases, modification of existing or addition of new requirements (ISO/IEC/IEEE
26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.38)
(3) determining the appropriate combination of processes, inputs, tools, techniques, outputs, and life cycle phases to
manner in which any selected issue is addressed in a particular project (INCOSE Systems Engineering Handbook, 4th
ed.) Note: Tailoring may be applied to various aspects of the project, including project documentation; processes and
activities performed in each life cycle stage; or the time and scope of reviews, analysis, and decision making, consistent
with applicable statutory requirements.
tailoring guideline. (1) instructions that enable an organization to adapt standard processes appropriately to meet
specific needs (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.1.11)
Note: Tailoring a process adapts the process description for a particular end. For example, a project creates its defined
process by tailoring the organization's set of standard processes to meet the objectives, constraints, and environment of
the project. The organization's set of standard processes is described at a general level that is not directly usable to
perform a process. Tailoring guidelines aid those who establish the defined processes for specific needs. Tailoring
guidelines describe what can and cannot be modified and identify process components that are candidates for
modification.
tangibility. (1) degree to which the tangible aspects of the IT service effectively communicate and support the service
(ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation
(SQuaRE)--Service quality models, 3.2.5)
target audience. (1) group of persons for whom information for use is intended by the supplier (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.42)
target entity. (1) fundamental thing of relevance to the user, about which information is kept, and which needs to be
measured (ISO/IEC 25021:2012 Software engineering--Software product Quality Requirements and Evaluation
(SQuaRE)--Quality measure elements, 4.17) Note: Possible synonyms of target entity are input to information product and
work product. Examples of target entities are architecture, contextual schema, conceptual and logical and physical data
models, data dictionary, document, data file, database management, relational database management system, form, and
presentation device. Target entities are precisely defined by properties. Examples of properties are attribute, element,
information, metadata, vocabulary, data format, data item, data value, information item, information item content, and data record.

target language. (1) language in which the output from a machine-aided translation process is represented (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) language into which source language content is translated (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.32) Syn: object language See also: source language

target machine. (1) computer on which a program is intended to execute (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) a computer being emulated by another computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: host machine

target node. (1) node associated with the end of an arc (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.18)

target of process. (1) system, software product or task executed by system or software product to which measurement or evaluation process is applied (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.39)

target of verification (TOV). (1) software or a set of software items or units to be verified, e.g. in terms of safety and security (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.26)

target process profile. (1) process profile specifying which process attributes are required and the rating necessary for each process attribute for a required process (ISO/IEC 33001:2015 Information technology--Process assessment--Concepts and terminology, 3.2.20)

target risk. (1) risk that is intended to be reached (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.3.13)

target system. (1) complete computing platform capable of running the target software (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.28) (2) system to be categorized, which can be an IT system and software, including service provided by IT system (ISO/IEC TR 12182:2015 Systems and software engineering--Framework for categorization of IT systems and software, and guide for applying it, 3.4) Note: consists of hardware resources and software resources installed on the hardware.

operating system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (6) smallest unit of work subject to management accountability; a well-defined work assignment for one or more project members (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (7) set or sequence of activities required to achieve a given goal (ISO/IEC 25023:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Measurement of system and software product quality, 4.12) (8) recommended action intended to contribute to the achievement of one or more outcomes of an architecture process (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.23) Note: Related tasks are usually grouped to form activities.

task behavior specification. (1) specification describing a concurrent task's interface, structure, timing characteristics, relative priority, errors detected, and task event sequencing logic (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

task completion. (1) timely event when for a specific task the total output string or, in case of a set of output strings, all parts are completely received by to the emulated user or another instance (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.2) Note: The time of task completion defines the end time of the preceding preparation time and the begin time of the execution time of the following task.

task interface. (1) input or output, events signaled (input or output), external inputs or outputs, or access to passive objects (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

task inversion. (1) optimization concept whereby the tasks in a system can be merged in a systematic way. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

task mode. (1) indication of whether the user's preparation time begins immediately with the task submission of the preceding task (value = 0, i.e., "NO WAIT") or begins when the preceding task has been completed (task completion) (value = 1, i.e., "WAIT") (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.21) Note: mode of "Dialog" or "Batch" in UNIX-based systems.

task priority criteria. (1) category of the task-structuring criteria addressing the relative importance of executing a given task (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

task structuring. (1) software design stage with the objective of structuring a concurrent application into concurrent tasks and defining the task interfaces (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

task submission. (1) timely event when the input string is completely submitted from the emulated user to the SUT and the execution of the task can start, regardless if the SUT starts the execution immediately or not (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.22) Note: Normally the task submission is defined internally by the submission of a special character (e.g. Carriage Return) or a character sequence at the end of the input string or at the end of several parts of the input string. Also it often happens that the task submission event is defined by the submission of the last character of any specified number characters in a string. For a classic batch task, the task submission is defined by the submission of the last character of the last string of the batch command sequence.

task type. (1) classification of tasks which is defined by the combination of (1) the activity type, or a set of activity types which are all belonging to an identical timeliness function and task mode (2) the timeliness function; the task mode


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright®, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.
In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available.
Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
(ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.23) Note: Emulated users submit only these types of tasks to the SUT.

task-clustering criteria. (1) category of the criteria addressing whether and how to group objects into concurrent tasks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

task-structuring criteria. (1) set of heuristics for helping a designer structure a system into concurrent tasks (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

taxonomy. (1) scheme that partitions a body of knowledge and defines the relationships among the pieces (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.32) (2) classification scheme for referencing profiles or sets of profiles unambiguously (ISO/IEC 29110-2-1:2015 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 2-1: Framework and taxonomy, 4.63) Note: It is used for classifying and understanding the body of knowledge.

TBD. (1) to be determined (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)

TBR. (1) to be resolved (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2) (2) to be revised (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)

TBS. (1) to be supplied (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.1) (2) to be specified (ISO/IEC/IEEE 29148:2018 Systems and software engineering-Life cycle processes-Requirements engineering, 4.2)

TCP. (1) Transmission Control Protocol (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 4)

TCPI. (1) to complete performance index (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)


TDD. (1) test-driven development (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.2)

TDP. (1) technical data package (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

team charter. (1) a document that records the team values, agreements, and operating guidelines, as well as establishing clear expectations regarding acceptable behavior by project team members (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

team management plan. (1) a component of the resource management plan that describes when and how team
members will be acquired and how long they will be needed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

team selection plan. (1) document specifying the qualifications, experience and training needs of project staff (ISO/IEC/IEEE 24765a:2011)

technical complexity adjustment. (1) a factor which attempts to take into account the influence on application size of technical and quality requirements, which can be used to derive the adjusted size (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) Note: Note that if this is done, the result is not the functional size.

technical complexity adjustment factors. (1) the set of 19 factors that are taken into account in the technical complexity adjustment (TCA) (ISO/IEC 20968:2002 Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual, 10) Note: Each factor has a degree of influence (DI) between 1 and 5.

technical contact. (1) person responsible for providing an information developer with technical information about a product or for checking the technical accuracy of drafts of information for users (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.24) Syn: subject-matter expert, SME

technical debt. (1) the deferred cost of work not done at an earlier point in the product life cycle (Software Extension to the PMBOK(R) Guide Fifth Edition)

technical independence. (1) of software quality assurance (SQA), situation in which the SQA effort uses personnel who are not involved in the development of the system or its elements (IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2)


technical manager. (1) stakeholder with responsibility for decisions relating to product content and quality achievement (ISO/IEC/IEEE 24748-5:2017 Systems and software engineering--Life cycle management--Part 5: Software development planning, 3.22) Note: Technical decisions include definition and tailoring of life cycle processes, design of measurement systems, and product implementation decisions.

technical performance measure (TPM). (1) measure used to assess design progress, compliance to performance requirements, and technical risks for critical performance parameters (ISO/IEC/IEEE 24765h:2019) (2) measure that defines attributes of a system element to determine how well a system or system element is satisfying or expected to satisfy a technical requirement or goal (INCOSE Systems Engineering Handbook, 4th ed.)

technical performance measurement. (1) collection and comparison of technical accomplishments during project execution to the specified level of service, key performance indicators, or planned technical events and accomplishments (ISO/IEC/IEEE 24765c:2014)

technical product quality requirement. (1) product quality requirement on its technically identified properties which are used in its development and maintenance processes (ISO/IEC 25030:2019 Systems and software engineering--
technical requirements. (1) requirements relating to the technology and environment, for the development, maintenance, support and execution of the software (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
technical review. (1) series of systems engineering activities conducted at logical transition points in a system life cycle, by which the progress of a program is assessed relative to its technical requirements using a mutually agreed-upon set of criteria (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1) (2) formal peer review of a work product by a team of technically qualified personnel that examines the suitability of the work product for its intended use and identifies discrepancies from specifications and standards (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.18) Note: Technical reviews can also provide recommendations of alternatives and examination of various alternatives.
technical standard. (1) standard that describes the characteristics of applying accumulated technical or management skills and methods in the creation of a product or performing a service (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
technique. (1) a defined systematic procedure employed by a human resource to perform an activity to produce a product or result or deliver a service, and that may employ one or more tools. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition) (2) methods and skills required to carry out a specific activity (ISO/IEC 25001:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Planning and management, 4.5) (3) technical or managerial procedure that aids in the evaluation and improvement of the software development process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
technology roadmap. (1) outline of required and anticipated changes in technologies, with expected dates, which will enable achievement or transformation of a product or product family (ISO/IEC 26560:2019 Software and systems engineering -- Tools and methods for product line product management, 3.6) See also: product roadmap
technology viewpoint. (1) viewpoint on an ODP system and its environment that focuses on the choice of technology in that system (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 4.1.1.5)
TEMP. (1) test and evaluation master plan (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)
template. (1) asset with parameters that can be used to construct an instantiated asset (ISO/IEC/IEEE 24765j:2021) (2) a partially complete document in a predefined format that provides a defined structure for collecting, organizing, and presenting information and data (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide -- Sixth Edition) (3) specification of the common features of a collection of &lt;X&gt;'s in sufficient detail that an &lt;X&gt; can be instantiated using it (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model:}
template class. (1) of an \&lt;&lt;\&gt;&gt;, the set of all \&lt;&lt;\&gt;&gt;'s satisfying an \&lt;&lt;\&gt;&gt; template type (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.23)

template type. (1) of an \&lt;&lt;\&gt;&gt;, a predicate defined in a template that holds for all the instantiations of the template and that expresses the requirements the instantiations of the template are intended to fulfill (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.22)

temporal clustering. (1) task-structuring criterion by which activities that are not sequentially dependent, but are activated by the same event are grouped into a task (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

temporal cohesion. (1) type of cohesion in which the tasks performed by a software module are all required at a particular phase of program execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: coincidental cohesion, communicational cohesion, functional cohesion, logical cohesion, procedural cohesion, sequential cohesion


terminal. (1) functional unit in a system or communication network at which data can be entered or retrieved (ISO/IEC 2382:2015 Information technology -- Vocabulary)

terminal symbol. (1) part of the hierarchical definition of a syntax that is not further decomposed in the hierarchy (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2)

terminating behavior. (1) behavior which breaks down a liaison and repudiates the corresponding contractual context and the corresponding contract (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.2.5) Syn: terminating behaviour

termination construct. (1) program construct that results in a halt or exit (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

termination deliver. (1) signal in the implicitly defined signal interface of a client computational object which has the same name and parameters as one of the terminations of an interrogation in the original operation interface (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.13)

termination submit. (1) signal in the implicitly defined signal interface of a server computational object which has the same name and parameters as one of the terminations of an interrogation in the original operation interface (ISO/IEC 14752:2000 Information technology -- Open Distributed Processing -- Protocol support for computational interactions, 3.3.14) See also: invocation submit, invocation deliver

terminology management system. (1) software tool specifically designed for collecting, maintaining, and accessing terminological data (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.34)
test. (1) activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4) (2) to conduct an activity as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) set of one or more test cases and procedures (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

test and evaluation document. (1) project document that describes the activities used to determine if the product meets the quality objectives stated in the quality management plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

test approach. (1) particular method that will be employed to pick the particular test case values (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: varies in specificity from very general (e.g., black box or white box) to very specific (e.g., minimum and maximum boundary values) See also: test practice

test basis. (1) body of knowledge used as the basis for the design of tests and test cases (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.47) Note: The test basis can take the form of documentation, such as a requirements specification, design specification, or module specification, but can also be an undocumented understanding of the required behavior.

test bed. (1) environment containing the hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

test case. (1) 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 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.3.31) (2) documentation specifying inputs, predicted results, and a set of execution conditions for a test item (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.31) (3) documentation specifying inputs, predicted results, and a set of execution conditions for a test item (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) set of test case preconditions, inputs (including actions, where applicable), and expected results, developed to drive the execution of a test item to meet test objectives, including correct implementation, error identification, checking quality, and other valued information (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.48) Note: A test case is the lowest level of test input (i.e. test cases are not made up of test cases) for the test subprocess for which it is intended

test case generator. (1) software tool that accepts as input source code, test criteria, specifications, or data structure definitions; uses these inputs to generate test input data; and, sometimes, determines expected results (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: test data generator, test generator


test class. (1) designated grouping of test cases (ISO/IEC/IEEE 24765:2017 Systems and software engineering--
Vocabulary

test completion process. (1) test management process for ensuring that useful test assets are made available for later use, test environments are left in a satisfactory condition, and the results of testing are recorded and communicated to relevant stakeholders (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.50)

test completion report. (1) report that summarizes the testing that was performed (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.51) Syn: test summary report

test condition. (1) testable aspect of a component or system, such as a function, transaction, feature, quality attribute, or structural element identified as a basis for testing (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.52)

test coverage. (1) degree, expressed as a percentage, to which specified test coverage items have been exercised by a test case or test cases (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.53)

test coverage item. (1) attribute or combination of attributes that is derived from one or more test conditions by using a test design technique that enables the measurement of the thoroughness of the test execution (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.54) Syn: coverage item See also: test item

test criteria. (1) criteria that a system or component must meet in order to pass a given test (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: acceptance criteria, pass/fail criteria

test data. (1) data created or selected to satisfy the input requirements for executing one or more test cases, which can be defined in the test plan, test case, or test procedure (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.34) (2) independent dataset used to provide an unbiased evaluation of the final, tuned machine learning model (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.75)


test design. (1) documentation specifying the details of the test approach for a system, software, or hardware feature or combination of features and identifying the associated tests (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.32) Note: commonly includes the organization of the tests into groups

test design and implementation process. (1) test process for deriving and specifying test cases and test procedures (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.57)

test design specification. (1) document specifying the features to be tested and their corresponding test conditions (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.58)

test design technique. (1) activities, concepts, processes, and patterns used to construct a test model that is used to identify test conditions for a test item, derive corresponding test coverage items, and subsequently derive or select test
test documentation. (1) documentation describing plans for, or results of, the testing of a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) collection of the documentation inherent to the testing activities (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.19) Note: Types include test case specification, test incident report, test log, test plan, test procedure, test report.

test driver. (1) software module used to invoke a module under test and, often, provide test inputs, control and monitor execution, and report test results (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: test harness

test effort. (1) activity of performing one or more testing tasks (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

test environment. (1) facilities, hardware, software, firmware, procedures, and documentation intended for or used to perform testing of software (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.60) Note: A test environment could contain multiple environments to accommodate specific test sub-processes, e.g., a unit test environment, a performance test environment.


test execution. (1) process of running a test on the test item, producing actual results (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.64)

test execution engine. (1) tool implemented in software and sometimes in hardware that can manipulate the test item to execute test cases (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.11)

test execution log. (1) document that records details of the execution of one or more test procedures (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.65)

test execution process. (1) dynamic test process for executing test procedures created in the test design and implementation process in the prepared test environment, and recording the results (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.68)


test harness. (1) scaffolding code written for the purpose of exercising lower level code when the higher-level code
that will ultimately exercise it is not yet available. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: test driver

test incident report. (1) document reporting on any event that occurs during the testing process which requires investigation. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

test incident reporting process. (1) dynamic test process for reporting to the relevant stakeholders’ issues requiring further action that were identified during the test execution process. (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.67)

test interface. (1) interface to the test item used to stimulate the test item, to get responses (e.g. actual results), or both. (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.13)

test interface layer. (1) lowest level of abstraction for keywords, which interacts with the test item directly and encapsulates the atomic (lowest level) interactions at the test interface. (ISO/IEC/IEEE 29119-5:2016 Software and systems engineering--Software testing--Part 5: Keyword-driven testing, 4.14)

test item. (1) work product that is an object of testing. (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.68) Syn: test object


test lead. (1) person who leads entire testing process. (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.39)


test log. (1) chronological record of relevant details about the execution of tests. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)


test management process. (1) test process containing the sub-processes that are required for the management of a test project. (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.71) See also: test monitoring and control process, test completion process

test model. (1) representation of a test item that is used during the test case design process. (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.27)

test monitoring and control process. (1) test management process for ensuring that testing is performed in accordance with a test plan and with organizational test specifications. (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.72)

test objective. (1) identified set of software features to be measured under specified conditions by comparing actual behavior with the required behavior. (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.9) (2) identified set of software


This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
characteristics to be measured under specified conditions by comparing actual behavior with the required behavior (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.20)

test oracle. (1) source of information for determining whether a test has passed or failed (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.76)

Note: The test oracle is often a specification used to generate expected results for individual test cases, but other sources can be used, such as comparing actual results with those of another similar program or system, or asking a human expert.

test oracle problem. (1) challenge of determining whether a test has passed or failed for a given set of test inputs and state (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.77)

test participant. (1) person who provides feedback and allows data collection to test that the information in the software documentation is sufficient to accomplish tasks correctly and form a conceptual understanding of the system (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.40)

test phase. (1) period of time in the software life cycle during which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) specific instantiation of test sub-process (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.52)

See also: test level

test plan. (1) document that describes the technical and management approach to be followed to test a system or component (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (2) document describing the scope, approach, resources, and schedule of intended testing activities (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation) (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1) (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.1) (3) plan that establishes detailed requirements, criteria, general methodology, responsibilities, and general planning for test and evaluation of a system (ISO/IEC 2388:2015 Information technology -- Vocabulary) (4) detailed description of test objectives to be achieved and the means and schedule for achieving them, organized to coordinate testing activities for some test item or set of test items (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.75)

Note: It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. Typical contents identify the items to be tested, tasks to be performed, responsibilities, schedules, and required resources for the testing activity.

test planning process. (1) test management process used to complete test planning and develop test plans (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.76)
test practice. (1) conceptual framework that can be applied to the organizational test process, the test management process, or the dynamic test process to facilitate testing (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.77) See also: test approach
test procedure specification. (1) document specifying a sequence of actions for the execution of a test (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) document specifying one or more test procedures, which are collections of test cases to be executed for a particular objective (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.79) Note: A test procedure specification for an automated test run is usually called a test script.
test process. (1) process that provides information on the quality of a software product (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.80) Note: often comprised of a number of activities, grouped into one or more test sub-processes
test protocol. (1) list of the steps to be followed in the test (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.41)
test readiness review (TRR). (1) review conducted to evaluate preliminary test results for one or more configuration items; to verify that the test procedures for each configuration item are complete, comply with test plans and descriptions, and satisfy test requirements; and to verify that a project is prepared to proceed to formal testing of the configuration items (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) review as in (1) for any hardware or software component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: code review, formal qualification review, design review, requirements review
test repeatability. (1) attribute of a test, indicating that the same results are produced each time the test is conducted (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)
test report. (1) document that describes the conduct and results of the testing carried out for a system or component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: test case specification, test completion report, test incident report, test item transmittal report, test log, test plan, test procedure
test result. (1) indication of whether or not a specific test case has passed or failed, i.e. if the actual result observed as test item output corresponds to the expected result or if deviations were observed (ISO/IEC/IEEE 29119-1:2013 Software engineering--Software testing--Part 1: Concepts and definitions, 4.72)
and systems engineering--Software testing--Part 1: Concepts and definitions, 4.82)

**test scenario.** (1) situation or setting for a test item used as the basis for generating test cases (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.78)

**test script.** (1) test procedure specification for manual or automated testing (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.83)

**test set.** (1) set of one or more test cases with a common constraint on their execution (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.84) (2) collection of test cases for the purpose of testing a specific test objective (ISO/IEC/IEEE 29119-2:2013 Software and systems engineering--Software testing--Part 2: Test processes, 4.62) Note: The test sets will typically reflect the feature sets, but they could contain test cases for a number of feature sets. Test cases for a test set could be selected based on the identified risks, test basis, retesting, or regression testing.

**test set architecture.** (1) nested relationships between sets of test cases that directly reflect the hierarchic decomposition of the test objectives (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

**test specification.** (1) complete documentation of the test design, test cases and test procedures for a specific test item (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.85) Note: A test specification could be detailed in one document, in a set of documents, or in other ways, for example, in a mixture of documents and database entries.

**test status report.** (1) report that provides information about the status of the testing that is being performed in a specified reporting period (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.86)

**test strategy.** (1) part of the Test Plan that describes the approach to testing for a specific test project or test sub-process or sub-processes (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.87) Note: The test strategy usually describes some or all of the following: the test practices used; the test subprocesses to be implemented; the retesting and regression testing to be employed; the test design techniques and corresponding test completion criteria to be used; test data; test environment and testing tool requirements; and expectations for test deliverables.

**test sub-process.** (1) test management and dynamic (and static) test processes used to perform a specific test level (e.g. system testing, acceptance testing) or test type (e.g. usability testing, performance testing) normally within the context of an overall test process for a test project (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.88) Note: Depending on the life cycle model used, test sub-processes are also typically called test phases, test levels, test stages or test tasks. Syn: test subprocess

**test target version.** (1) specific version of test target which is used for one-time execution of Dynamic Test Execution or Code Analysis (ISO/IEC 30130:2016 Software engineering --Capabilities of software testing tools)

**test traceability matrix.** (1) document, spreadsheet, or other automated tool used to identify related items in documentation and software, such as requirements with associated tests (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.90) Syn: verification cross reference matrix,
requirements test matrix, requirements verification table See also: traceability matrix

test type. (1) group of testing activities that are focused on specific quality characteristics (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.91)

test unit. (1) set of one or more computer program modules together with associated control data (for example, tables), usage procedures, and operating procedures that satisfy the following conditions: (a) All modules are from a single computer program; (b) At least one of the new or changed modules in the set has not completed the unit test; (c) The set of modules together with its associated data and procedures are the sole object of a testing process. (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

testability. (1) extent to which an objective and feasible test can be designed to determine whether a requirement is met (ISO/IEC/IEEE 24765g:2018) (2) degree of effectiveness and efficiency with which test criteria can be established for a system, product, or component and tests can be performed to determine whether those criteria have been met (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.7.5) (3) degree of effectiveness and efficiency with which test criteria can be established for an IT service and tests can be performed to determine whether those criteria have been met (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.8.3)

testing. (1) activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.42) (2) process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.23) (3) set of activities conducted to facilitate discovery or evaluation of properties of one or more test items (ISO/IEC/IEEE 29119:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.92)

testing description. (1) description of the test execution conditions (i.e. test procedure) (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.11) (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.11)

testing task iteration. (1) testing task that is re-performed during maintenance after having been originally performed during development (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

testing tool. (1) specific or generic tool which is used for test execution and test management such as test results recording, test results display, test results interpretation, generation of test data, generation of test procedure, generation of test scripts, test modelling (ISO/IEC 30130:2016 Software engineering --Capabilities of software testing tools)

testware. (1) products produced by the testing effort (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) artifacts produced during the test process required to plan, design, and execute tests (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)
text. (1) data in the form of characters, symbols, words, phrases, paragraphs, sentences, tables, or other character arrangements, intended to convey a meaning, and whose interpretation is essentially based upon the reader's knowledge of some natural language or artificial language (ISO/IEC 2382:2015 Information technology -- Vocabulary)

text editor. (1) computer program, often part of a word processing system, that allows a user to enter, alter, and view text (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: editor

text page. (1) model page that contains textual material related to a specific diagram (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.102)

text processing. (1) data processing operations on text, such as entering, editing, merging, retrieving, storing, displaying, or printing (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: word processing


theme. (1) user stories associated by a common factor, such as functionality, data source, or security level (Software Extension to the PMBOK(R) Guide Fifth Edition)

think time. (1) elapsed time between the end of a prompt or message generated by an interactive system and the beginning of a human user's response (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: port-to-port time, response time, turnaround time

third normal form. (1) result of a normalization process that transforms groups of data so that each non-key attribute does not depend on any other non-key attribute (ISO/IEC/IEEE 24765a:2011)

third party. (1) person or body that is recognized as being independent of the parties involved, as concerns the issue in question (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.25) Syn: third-party

thrashing. (1) state in which a computer system is expending most or all of its resources on overhead operations, such as swapping data between main and auxiliary storage, rather than on intended computing functions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

thread. (1) chain of actions, where at least one object participates in all the actions of the chain (ISO/IEC 10746-1:1998 Information technology -- Open Distributed Processing -- Reference model: Overview, 13.1.2)

threat. (1) state of the system or system environment which can lead to adverse effects (ISO/IEC/IEEE 24765a:2011) (2) a risk that would have a negative effect on one or more project objectives (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

threat modeling. (1) systematic exploration technique to expose any circumstance or event having the potential to cause harm to a system in the form of destruction, disclosure, modification of data, or denial of service (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: It results in a vulnerability assessment.

three-address instruction. (1) a computer instruction that contains three address fields (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-address instruction, two-address instruction, four-address instruction, zero-address instruction

three-plus-one address instruction. (1) computer instruction that contains four address fields, the fourth containing the address of the instruction to be executed next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: for example, See also: one-plus-one address instruction, two-plus-one address instruction, four-plus-one address instruction

three-point estimate. (1) a technique used to estimate cost or duration by applying an average of optimistic, pessimistic, and most likely estimates when there is uncertainty with the individual activity estimates (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

threshold. (1) a predetermined value of a measurable project variable that represents a limit that requires action to be taken if it is reached (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

threshold coverage. (1) proportion of neurons exceeding a threshold activation value divided by the total number of neurons in the neural network (normally expressed as a percentage) for a set of tests (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.79) Note: A threshold activation value between 0 and 1 is chosen as the threshold value.

throughput. (1) amount of work that can be performed by a computer system or component in a given period of time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) rate (i.e., the average number per time unit with respect to the rating interval) of all tasks of a task type submitted to the SUT (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.24) Note: Usually throughput is defined by the rate of terminated tasks during a period of time.

throughput rating value. (1) quotient (corresponding to the j-th task type) of the (actual) throughput and the throughput reference value (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.25)

throughput reference value. (1) minimum throughput required by the set of emulated users (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.26)

thumbnail. (1) miniature image file displayed for quick identification of a larger image or video file (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.24)

tier. (1) grouping of process definitions (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.50)

TIM. (1) technical interchange meeting (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

time. (1) in decreasing order of resolution, CPU execution time, elapsed time (i.e., wall clock time), or calendar time
time and material (T&M) contract. (1) a type of contract that is a hybrid contractual arrangement containing aspects of both cost-reimbursable and fixed-price contracts. 

Time and material contracts resemble cost-reimbursable type arrangements in that they have no definitive end, because the full value of the arrangement is not defined at the time of the award. Thus, time and material contracts can grow in contract value as if they were cost-reimbursable-type arrangements. Conversely, time and material arrangements can also resemble fixed-price arrangements. For example, the unit rates are preset by agreement of the acquirer and supplier.

time behavior. (1) degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements. 

Time behavior

time class. (1) time limit, combined with a relative frequency corresponding to the ratio of the number of tasks (of a specific task type), with an execution time less than or equal to the corresponding time limit, to the total number of tasks (of that particular task type), used for comparison with the execution time of a task (of that particular task type).

Time class

time out. (1) condition that occurs when a predetermined amount of time elapses without the occurrence of an expected event.

time Petri net. (1) Petri net with time where timing constraints are associated with transitions.

Time Petri net

time sharing. (1) mode of operation that permits two or more users to execute computer programs concurrently on the same computer system by interleaving the execution of their program.

Time sharing

time slicing. (1) mode of operation in which two or more processes are each assigned a small, fixed amount of continuous processing time on the same processor, and the processes execute in a round-robin manner, each for its allotted time, until all are completed.

Time slicing

time-boxed. (1) having a prescribed duration limit for a project task.

Time-boxed

time-critical task. (1) task that must meet a hard deadline.

Time-critical task

time-scaled schedule network diagram. (1) project schedule network diagram drawn in such a way that the positioning and length of the schedule activity represents its duration.

Time-scaled schedule network diagram
technological--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.6.1)
timeliness function. (1) description of the user requirements with respect to the execution times of tasks of a
specific task type (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-
based software systems, 4.29) Note: It consists of one or more time classes.
timeliness rating value. (1) quotient (corresponding to the j-th task type) of the timely throughput and the total
throughput (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based
software systems, 4.28)
timely throughput. (1) throughput of all of those tasks whose execution times are accepted with respect to the
timeliness function (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based
software systems, 4.3)
timer event. (1) stimulus used to periodically activate a task (ISO/IEC/IEEE 24765:2017 Systems and software
engineering-Vocabulary)
timesharing. (1) operating technique of a data processing system that provides for the interleaving in time of two or
more processes in one processor (ISO/IEC 2382:2015 Information technology -- Vocabulary)
timing. (1) process of estimating or measuring the amount of execution time required for a software system or
component (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: sizing
timing analysis. (1) method to evaluate the time sequence of logic signals, or the speed of a digital circuit
timing analyzer. (1) software tool that estimates or measures the execution time of a computer program or portion of
a computer program, either by summing the execution times of the instructions along specified paths or by inserting
probes at specified points in the program and measuring the execution time between probes (ISO/IEC/IEEE 24765:2017
Systems and software engineering-Vocabulary)
timing diagram. (1) diagram showing the time-ordered execution sequence of a group of tasks (ISO/IEC/IEEE
24765:2017 Systems and software engineering-Vocabulary)
TINA. (1) Telecommunication Information Networking Architecture (ISO/IEC 10746-1:1998 Information technology --
Open Distributed Processing -- Reference model: Overview)
tinderbox. (1) automated build and regression-testing tool (ISO/IEC/IEEE 24765:2017 Systems and software
engineering-Vocabulary) Note: A tinderbox will typically fetch on a regular basis the latest versions of the software from
each supported branch, build it for the different platforms, and report the results from the build and the regression tests.
TLS. (1) Transport Layer Security (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and
management of websites for systems, software, and services information, 5)
software engineering--Requirements for managers of information for users of systems, software, and services, 3.2)
TMRR. (1) technology maturation and risk reduction (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and
Audits on Defense Programs, 3.2)
to-complete performance index (TCPI). (1) a measure of the cost performance that must be achieved with the
remaining resources in order to meet a specified management goal, expressed as the ratio of the cost to finish the outstanding work to the remaining budget (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

TOC. (1) total ownership cost (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

TOE. (1) target of evaluation (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.26)

token. (1) terminal symbol (ISO/IEC 15475-2:2002 Information technology -- CDIF transfer format -- Part 2: Syntax SYNTAX.1, 6.1)


tolerance. (1) the quantified description of acceptable variation for a quality requirement (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

tool. (1) software product that provides support for software and system life cycle processes (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) (2) something tangible, such as a template or software program, used in performing an activity to produce a product or result. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

tool-specific information. (1) information associated with an object of a net graph or with the net graph itself that is specific to a particular tool and is not meant to be used by other tools (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.1.19) Syn: tool specific information

toolbox. (1) set of tools completing each other in terms of capabilities, to cover a larger area of their intended use (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.29) Syn: tool suite

top box. (1) box in the A-0 context diagram that models the top-level function of an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.103)

top management. (1) person or group of people who direct and control an organization at the highest level (ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems--Requirements, 3.55) Note: Top management has the power to delegate authority and provide resources within the organization. In a very small enterprise, the concept of top management may not be applicable.

top-down. (1) pertaining to an activity that starts with the highest-level component of a hierarchy and proceeds through progressively lower-levels (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) pertaining to a method or procedure that starts at the highest level of abstraction and proceeds towards the lowest level (ISO/IEC 2382:2015 Information technology -- Vocabulary) See also: bottom-up, critical piece first

top-down design. (1) design approach in which a system's functionality is decomposed from high-level concepts into lower-level pieces (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) process of designing a
system by identifying its major components, decomposing them into their low-level components, and iterating until the desired level of detail is achieved (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: top-down decomposition

top-level function. (1) function modeled by the single box in the A-0 context diagram of an IDEF0 model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.104)

topic. (1) small part of a document that deals with a single subject (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.50) (2) unit of information that deals with a single subject (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.43) (3) standalone item of information that answers a single user question (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 4.33) (4) coherent part of an information product with a heading that deals with a single subject (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.43) Note: In printed documentation, a topic is equivalent to a section (heading; subheading) and its content. In onscreen documentation, a topic consists of a title (heading) and information about a subject (typically, a task or a concept or reference information). For on-screen documentation, the system can present a topic without user intervention.

tornado diagram. (1) a special type of bar chart used in sensitivity analysis for comparing the relative importance of the variables (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

total. (1) complete mapping (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.196) Note: The mapping M from a set D to a set R is total if for every X in D, there is at least one Y in R and pair [X,Y] in M. A property of a class is total, meaning that it will have a value for every instance of the class, unless it is explicitly declared partial. See also: partial, mandatory, mapping completeness

total cluster. (1) subclass cluster in which each instance of a superclass must be an instance of at least one of the subclasses of the cluster (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.197) Syn: complete cluster See also: incomplete cluster, partial cluster, superclass

total correctness. (1) in proof of correctness, a designation indicating that a program's output assertions follow logically from its input assertions and processing steps (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: partial correctness

total float (TF). (1) the amount of time that a schedule activity can be delayed or extended from its early start date without delaying the project finish date or violating a schedule constraint (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

total quality management (TQM). (1) holistic approach to quality improvement in all life-cycle phases (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

TOV. (1) target of verification (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.26)

TP. (1) Transaction Processing (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing --

TPU. (1) timer pulse unit (ISO/IEC/IEEE 24765d:2015)

TQM. (1) Total Quality Management (ISO/IEC/IEEE 24765c:2014)

TR. (1) technical requirements (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 4) (2) ODP Type Repository (ISO/IEC 14769:2001 Information technology -- Open Distributed Processing -- Type Repository Function, 4)

TRA. (1) threat and risk assessment (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)

trace. (1) record of the execution of a computer program, showing the sequence of instructions executed, the names and values of variables, or both (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to produce a record as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) to establish a relationship between two or more products of the development process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (4) record of an object's interactions, from its initial state to some other state (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.7) Note: Types include execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace.

trace link. (1) association between two trace artifacts or between an element of variability model and a development artifact (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 3.6)

trace link semantics. (1) purpose or meaning of the trace link, specified in the trace link types (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 3.7)

traceability. (1) degree to which a relationship can be established among two or more logical entities, especially entities having a predecessor-successor or master-subordinate relationship to one another, such as requirements, system elements, verifications, or tasks (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.69) (2) discernible association among two or more logical entities, such as requirements, system elements, verifications, or tasks (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.71) (3) degree to which each element in a software development product establishes its reason for existing (ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.14) (4) degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (5) degree to which the IT service outcomes can be traced to or from the user needs (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality
traceability matrix. (1) matrix that records the relationship between two or more products of the development process (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

traceability with different abstraction level. (1) level of detail for established traceability (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 3.5)

traceable. (1) having components whose origin can be determined (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.25)

trade secret. (1) formula, process, design, or intellectual property that is protected by non-disclosure (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

trade study. (1) evaluation of alternatives, based on criteria and systematic analysis, to select the best alternative for attaining determined objectives (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)


trade-off analysis. (1) analytical evaluation of design options/alternatives against performance, design-to-cost objectives, and life cycle quality factors (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

trademark. (1) symbol, word, or phrase used to denote a particular source of goods or services (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)

trading. (1) interaction between objects in which information about new or potential contracts is exchanged via a third-party object (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.5.5)

trailer. (1) Identification or control information placed at the end of a file or message (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: header (2)

trailing decision. (1) loop control that is executed after the loop body (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: leading decision UNTIL

training. (1) provision of formal and informal learning activities (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: includes in-class instruction, informal mentoring, Web-based tutorials, guided self-study, and formalized on-the-job exercises. The learning options selected for each situation are based on an assessment of the performance gap to be addressed and resources.

training data. (1) dataset used to train a machine learning model (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.80)

transaction. (1) in software engineering, a data element, control element, signal, event, or change of state that causes, triggers, or initiates an action or sequence of actions (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary)
**transaction analysis.** (1) software development technique in which the structure of a system is derived from analyzing the transactions that the system is required to process (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: transaction-centered design See also: data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transform analysis

**transaction file.** (1) a temporary data file (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: it is read one time only and its data is consumed.

**transaction matrix.** (1) matrix that identifies possible requests for database access and relates each request to information categories or elements in the database (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary)

**transaction schema.** (1) dynamic schema and an invariant schema defining transactions and their dependencies (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.7.1.1)

**transaction transparency.** (1) distribution transparency which masks coordination of activities amongst a configuration of objects to achieve consistency (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 16.8.1.1)

**transactional function.** (1) elementary process that provides functionality to the user to process data (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.49) (2) a transaction (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: A succession of actions which the user sees as a single work unit. FPA assigns each transactional function a type and therefore distinguishes between the following types: external input, external output, and external inquiry.

**transactional function type.** (1) one of three categories that FPA assigns to a transactional function external input, external output, and external inquiry (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

**transfer.** (1) to send data from one place and receive it at another (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) (2) to relinquish control by one process and assume it at another, either with or without expectation of return (ISO/IEC 24765:2017 Systems and software engineering-Vocabulary) See also: jump

**transfer file.** (1) file containing data to be interchanged (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.1) Note: It is made up of a header and a number of components. Components contain either data or data definition data.

**transfer of an entitlement.** (1) process of assigning a given entitlement to a separate legal entity (ISO/IEC 19770-3:2016 Information technology--IT asset management--Part 3: Entitlement schema, 3.1.41) Note: Transfers often occur
when a large organization divests a part of itself into a separate legal entity. The Ent enables the recording of entitlement transfers. Transfers are in accordance with the contractual terms and conditions between the software licensor and end-user.

**transformation analysis.** (1) software development technique in which the structure of a system is derived from analyzing the flow of data through the system and the transformations that must be performed on the data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: transformation analysis, transform-centered design See also: data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transaction analysis

**transient error.** (1) error that occurs once, or at unpredictable intervals (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: intermittent fault, random failure

**transition.** (1) activities involved in moving a new or changed service, system, or component to or from an environment (ISO/IEC/IEEE 24765c:2014) (2) change from one state to another state or the same state (ISO/IEC 11411:1995 Information technology -- Representation for human communication of state transition of software, 2.2) (3) node of a net, usually represented by a rectangle (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 3.37) (4) node of a net, taken from the transition kind, and represented by a rectangle in the net graph. (ISO/IEC 15909-1:2019 Systems and software engineering--High-level Petri nets--Part 1: Concepts, definitions and graphical notation, 2.1.26) (5) [product line] switching to product line engineering through the right procedures, so as to achieve business objectives that lead an organization to product line engineering (ISO/IEC 26561:2019 Software systems engineering--Methods and tools for product line technical probe, 3.8) (6) switching to product line engineering from single system engineering (ISO/IEC 26562:2019 Software and systems engineering--Methods and tools for product line transition management, 3.4)

**transition plan.** (1) [product line] plan that describes product line transition strategy, resources required, responsibilities, detailed transition processes, major changes in product development and success measures for operationalizing the product line transition (ISO/IEC 26562:2019 Software and systems engineering--Methods and tools for product line transition management, 3.5)

**transition scenario.** (1) [product line] scenario including who, what, in what procedures and orders, and how to do their roles and responsibilities for deploying product line transition strategy (ISO/IEC 26562:2019 Software and systems engineering--Methods and tools for product line transition management, 3.6)

**transition strategy.** (1) [product line] set of plans intended to switch to product line engineering (ISO/IEC 26562:2019 Software and systems engineering--Methods and tools for product line transition management, 3.7)

**translator.** (1) computer program that transforms a sequence of statements expressed in one language into an equivalent sequence of statements expressed in another language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: assembler, compiler

**transparency.** (1) level of accessibility to the algorithm and data used by the artificial intelligence-based system (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.81)

**trap.** (1) conditional jump to an exception or interrupt handling routine, often automatically activated by hardware, with
the location from which the jump occurred recorded (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) to perform the operation in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**tree diagram.** (1) systematic diagram of a decomposition hierarchy used to visualize a systematic set of rules as parent-to-child relationships (ISO/IEC/IEEE 24765h:2019)

**tree-structured chart.** (1) chart depicting program constructs defined in ISO/IEC 8631 and having the structure of a tree (ISO/IEC 14568:1997 Information technology -- DXL: Diagram eXchange Language for tree-structured charts, 3.1.1)

**trend analysis.** (1) an analytical technique that uses mathematical models to forecast future outcomes based on historical results (ISO/IEC/IEEE 24765h:2019) Note: It is a method of determining the variance from a baseline of a budget, cost, schedule, or scope parameter by using prior progress reporting periods’ data and projecting how much that parameter’s variance from baseline might be at some future point in the project if no changes are made in executing the project.

**trigger condition.** (1) an event or situation that indicates that a risk is about to occur (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**triggering event.** (1) event (something that happens) that causes a functional user of the piece of software to initiate (trigger) one or more functional processes (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.25) Syn: triggering event type

**triple constraint.** (1) framework for evaluating competing demands, such as schedule, cost, and quality (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: The triple constraint is often depicted as a triangle where one of the sides or one of the corners represents one of the parameters being managed by the project team.

**TRR.** (1) test readiness review (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2)

**true negative.** (1) correct reporting of a failure when it is a failure (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.82) See also: false negative, true positive,

**true positive.** (1) correct reporting of a pass when it is a pass (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.83) See also: false positive, true negative

**trunk.** (1) software’s main line of development; the main starting point of most branches (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: One can often distinguish the trunk from other branches by the version numbers used for identifying its files, which are shorter than those of all other branches.

**trust.** (1) degree to which a user or other stakeholder has confidence that a product or system will behave as intended (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.3.2)

**trustworthy data.** (1) data and related information that is accurate, complete, relevant, readily understood by and available to those authorized users who need it to complete a task (ISO/IEC 19770-1:2017 Information technology -- IT
asset management -- Part 1: IT asset management systems--Requirements, 3.56

**tunnel notation.** (1) pair of short shallow arcs, resembling a pair of left and right parentheses characters, that bracket the arrowhead or the arrowtail of an arrow segment *(IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.107)*

**tunneled arrow.** (1) arrow left undrawn between its attachment to an ancestral box and its appearance as a boundary arrow on some hierarchically consecutive descendent diagram *(IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.105)*

**tunneling.** (1) act of applying tunnel notation to an arrow segment *(IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.106)*

tuple. (1) set of fields or data items *(ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.37)* Note: Tuple can be used in place of record.

turnaround time. (1) elapsed time between the submission of a job to a batch processing system and the return of completed output *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: port-to-port time, response time, think time

turnkey. (1) pertaining to a hardware or software system delivered in a complete, operational state *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)*

turnkey system. (1) data processing system that is ready to use when installed and supplied to the user in a ready-to-run condition, possibly customized to a specific user or application *(ISO/IEC 2382:2015 Information technology -- Vocabulary)* Note: Some preparatory work on the user's data can be required.

tutorial. (1) instructional procedure in which the user exercises system functions, often using sample data that is supplied with the system or information for users *(ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.26)*

two-address instruction. (1) computer instruction that contains two address fields *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Syn: double-operand instruction See also: one-address instruction, three-address instruction, four-address instruction, zero-address instruction

two-level address. (1) indirect address that specifies the storage location containing the address of the desired operand *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* See also: n-level address

two-level encoding. (1) microprogramming technique in which different microoperations can be encoded identically into the same field of a microinstruction, and the one that is executed depends upon the value in another field internal or external to the microinstruction *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Syn: two level encoding See also: bit steering, residual control, single-level encoding

two-phase acquisition. (1) segmenting a project into an early phase that focuses on gathering requirements, addressing major risks, and project planning; and a later phase that completes the project if the outcome of the first phase is favorable *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: The final decision on whether to do the full project is deferred from the point when the uncertainties are the greatest (the beginning) to a point where the uncertainties are significantly reduced.

two-plus-one address instruction. (1) computer instruction that contains three address fields, the third
containing the address of the instruction to be executed next (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-plus-one address instruction, three-plus-one address instruction, four-plus-one address instruction

type. (1) identifiable entity with an associated predicate (a single-argument mathematical function with a Boolean result) defined over entities (ISO/IEC 19500-1:2012 Information technology--Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 1: Interfaces, 5.3.4) (2) of an <X>, a predicate characterizing a collection of <X>s (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 9.9) See also: class


UDF. (1) unit development folder (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: software development file


unambiguous. (1) described in terms that only allow a single interpretation, aided, if necessary, by a definition (ISO/IEC/IEEE 15289:2019 Systems and software engineering--Content of life-cycle information items (documentation), 5.26)

unanimity. (1) agreement by everyone in the group on a single course of action (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

unbinding behavior. (1) behavior that terminates a binding (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 13.5.4)

unbundle. (1) separation of arrow meanings, expressed by branching arrow segments (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.108) Note: That is, the separation of object types from an object type set.

uncertainty. (1) result of not having accurate or sufficient knowledge of a situation (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) state, even partial, of deficiency of information related to understanding or knowledge of an event, its consequence, or likelihood (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.40) Note: often the root cause of a risk factor. In business decision making, uncertainty refers to unquantified variation; the probabilities of the variations cannot be used in the decision analysis. See also: risk

unconditional jump. (1) jump that takes place regardless of execution conditions (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: conditional jump

underfitting. (1) generation of a machine learning model that does not reflect the underlying trend of the training data, resulting in a model that finds it difficult to make accurate predictions (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.84)
underflow exception. (1) exception that occurs when the result of an arithmetic operation is too small a fraction to be represented by the storage location designated to receive it (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: addressing exception, data exception, operation exception, overflow exception, protection exception

understandability. (1) ease with which a system can be comprehended at both the system-organizational and detailed-statement levels (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Understandability has to do with the system’s coherence at a more general level than readability does.

undirected graph. (1) graph (sense 2) in which no direction is implied in the internode connections (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: directed graph

unidimensionality. (1) existence of a single trait or construct underlying a set of measures (ISO/IEC 33003:2015 Information technology--Process assessment--Requirements for process measurement frameworks, 3.15)


uniform resource identifier (URI). (1) compact sequence of characters that identifies an abstract or physical resource available on the Internet (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.51) Note: The syntax used for URIs is defined in IETF RFC 3986.

Uniform Resource Locator (URL). (1) mechanism for identifying resources on the Internet (such as web pages) by specifying the address of the resource and the access protocol used (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.25) Note: The term as specified by the IETF is uniform resource identifier (URI) of which URL is a subset.

unintended user. (1) person whom an autonomous/intelligent system (A/IS) creator does not intend an A/IS to be used by, but who nonetheless interacts with the A/IS (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)

unique function. (1) a function that differs in form and/or logical processing from every other function provided by a certain application (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis)

uniqueness constraint. (1) constraint stating that no two distinct instances of a class agree on the values of all the properties that are named in the uniqueness constraint (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFObject), 3.1.199)

unit (of measure). (1) particular quantity defined and adopted by convention, with which other quantities of the same
kind are compared in order to express their magnitude relative to that quantity (ISO/IEC 25021:2012 Software
engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Quality measure elements, 4.18) Note:
Only quantities expressed in the same units of measurement are directly comparable. Examples of units include the
number of faults and the number of failures. Hour and meter are also units of measure. Units of measurement have
conventionally assigned names and symbols. See also: unit of measurement

unit of measurement. (1) particular quantity defined and adopted by convention, with which other quantities of the
same kind are compared in order to express their magnitudes relative to that quantity (ISO/IEC 19761:2011 Software
engineering--Measurement process, 3.39) Note: Units of measurement have conventionally assigned names and
symbols. See also: unit (of measure)

unit requirements documentation. (1) license for software use as originally purchased or procured, and which
can typically be linked directly to purchase records (ISO/IEC/IEEE 24765:2017 Systems and software engineering--
Vocabulary)

unit test. (1) testing of individual routines and modules by the developer or an independent tester (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) test of individual programs or modules in order to ensure
that there are no analysis or programming errors (ISO/IEC 2382:2015 Information technology -- Vocabulary) (3) test of
individual hardware or software units or groups of related units (ISO/IEC/IEEE 24765:2017 Systems and software
engineering-Vocabulary)

unit test framework. (1) environment that facilitates unit testing (ISO/IEC/IEEE 24765:2017 Systems and software
engineering-Vocabulary)

universal asynchronous receiver/transmitter (UART). (1) electronic unit used for serial communications
that translates data between parallel and serial forms (ISO/IEC/IEEE 24765d:2015) Note: The data format and
transmission speeds are configurable. Commonly part of a microcontroller.

universal serial bus (USB). (1) serial communication interface with two data lines and two power lines between a

unpack. (1) to recover the original form of one or more data items from packed data (ISO/IEC/IEEE 24765:2017
Systems and software engineering-Vocabulary) See also: pack

unscripted testing. (1) dynamic testing in which the tester's actions are not prescribed by written instructions in a
definitions, 4.94) UNSP-SC. (1) United Nations standard products and services code (ISO/IEC 19770-3:2016 Information technology--IT
asset management--Part 3: Entitlement schema, 3.2)

unspecialize. (1) change by an instance from being an instance of its current subclass within a cluster to being an
Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.200) See also: respecialize, specialize

Material reprinted with permission from Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK) Guide - Sixth
Edition, 2017. Copyright and all rights reserved.

PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
unstratified language. (1) language that can be used as its own metalanguage (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

unsupervised learning. (1) task of learning a function that maps unlabeled input data to a latent representation (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.85) See also: supervised learning

UNTIL. (1) single-entry, single-exit loop, in which the loop control is executed after the loop body (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: post-tested iteration See also: closed loop, WHILE, trailing decision

unwind. (1) in programming, to state explicitly and in full all of the instructions involved in multiple executions of a loop (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: straight-line coding

UOD. (1) Universe of Discourse (ISO/IEC 19793:2015 Information technology -- Open Distributed Processing -- Use of UML for ODP system specifications, 4)

up. (1) pertaining to a system or component that is operational and in service (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Such a system is either busy or idle. See also: busy, down, idle

up time. (1) period of time during which a system or component is operational and in service; that is, the sum of busy time and idle time (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: down time, busy time, idle time, mean time between failures, set-up time

updatable argument. (1) designation given to an operation argument that identifies an instance to which a request can be sent that will change the state of the instance (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.201) Note: An argument not designated as "updatable" means that no requests can be sent that can change the state of the instance identified by the argument.

update. (1) modification to any deliverable, project management plan component, or project document that is not under formal change control (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

upload. (1) to transfer programs or data from a connected computer to a computer with greater resources (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: typically, from a personal computer to a server

upward compatible. (1) pertaining to hardware or software that is compatible with a later or more complex version of itself (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: downward compatible

upward compression. (1) in software design, a form of demodularization in which a subordinate module is copied inline into the body of a superordinate module (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: lateral compression, downward compression

UR. (1) user requirements (ISO/IEC TR 14143-4:2002 Information technology -- Software measurement -- Functional size measurement -- Part 4: Reference model, 4)

URa. (1) utilize benchmarking results activity (ISO/IEC 29155-2:2013 Systems and software engineering--Information technology project performance benchmarking framework--Part 2: Requirements for benchmarking, 4)

URL. (1) uniform resource identifier (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.51)

URL. (1) uniform resource locator (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and
management of websites for systems, software, and services information)

**usability.** (1) extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use *(ISO/IEC 25064:2013 Systems and software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: User needs report, 4.16)* (2) degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use *(ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.4)* (3) extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use *(IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.44)* (4) degree to which an IT service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use *(ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.2)* Note: Usability can either be specified or measured as a product quality characteristic in terms of its subcharacteristics, or specified or measured directly by measures that are a subset of quality in use. See also: reusability

**usability analyst.** (1) person who observes users performing tasks using the software and documentation and records the actions the user took, problems the user encountered, and comments the user made during the test; and interprets these records to evaluate the results of the testing *(ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of information for users, 3.45)*

**usability defect.** (1) product attribute that leads to a mismatch between user intentions or user actions and the system attributes and behavior *(ISO/IEC 25066:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for Usability--Evaluation Report, 3.17)*

**usability finding.** (1) identified usability defect, usability problem, or positive usability-related attribute *(ISO/IEC 25066:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for Usability--Evaluation Report, 3.18)*

**usability inspection.** (1) evaluation based on the considered judgment of evaluators who examine the usability-related aspects of an interface with respect to specified criteria *(ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.16)* Note: Usability inspection is the generic term for several methods, including but not limited to heuristic evaluation, cognitive walkthroughs, standards inspection, pluralistic walkthroughs, and consistency inspections. The inspectors can include usability specialists, developers, end users or other types of professionals. The evaluative criteria can include good practice and/or documented principles, guidelines, requirements or standards. The evaluation can be conducted with or without the help of referenced documents.

**usability laboratory.** (1) typically, a suite of evaluation and observation rooms fitted with video and audio equipment for recording user responses *(ISO/IEC/IEEE 24765a:2011)*

**usability objective.** (1) stated level of usability expressed in terms of effectiveness, efficiency and satisfaction in a
specified context of use which can be verified (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.15) Note: Usability objectives can be stated as user requirements, in which case the level to be achieved is a usability requirement, or they can be stated as desired target levels, depending on their use in design and evaluation.

**usability problem.** (1) situation during use resulting in poor effectiveness, efficiency or satisfaction (ISO/IEC 25066:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for Usability--Evaluation Report, 3.19)

**usability test.** (1) test to determine whether an implemented system fulfils its functional purpose as determined by its users (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: fitness-for-use test See also: usability testing

**usability walkthrough.** (1) usability evaluation in which one or more evaluators step through a scenario playing the role of a user and identifying usability problems associated with successful completion of the scenario (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.18) Note: The evaluators can include usability specialists, developers, end users, or other types of professionals.

**usage mode.** (1) primary manner in which the documentation developer expects the document to be used (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.53)


**use.** (1) activity that the user may perform with or on the product during its whole life cycle (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.45)

**use case.** (1) in UML, a complete task of a system that provides a measurable result of value for an actor (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) description of behavioral requirements of a system and its interaction with a user (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 4.15) (3) specification of a sequence of actions, including variants, that a system (or other entity) can perform, interacting with actors of the system (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.31) Note: More formally, a use case defines a set of use case instances or scenarios.

**use case diagram.** (1) UML diagram that shows actors, use cases, and their relationships (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**use case model.** (1) model that describes a system’s functional requirements in terms of use cases (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

**use case specification.** (1) document that describes a use case (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: A use case specification’s fundamental parts are the use case name, brief description, precondition, basic flow, postcondition, and alternate flow.

**use error.** (1) user action or lack of user action while using the interactive system that leads to a different result than that intended by the manufacturer or expected by the user (ISO/IEC 25066:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for Usability--
Evaluation Report, 3.20) (2) act or omission of an act that results in a different system response than intended by the manufacturer or expected by the user (ISO/IEC 25022:2016, Systems and software engineering -- Systems and software quality requirements and evaluation (SQuaRE) -- Measurement of quality in use, 4.25) Note: Use errors can result from a mismatch between the characteristics of the user, user interface, task, or use environment. Use error includes the inability of the user to complete a task. Users might be aware or unaware that a use error has occurred. A malfunction of an interactive system that causes an unexpected result is not considered a use error. An unexpected physiological response of the patient is not by itself considered a use error.

use of IT. (1) planning, design, development, deployment, operation, management, and application of IT to meet the needs of the organization (ISO/IEC/IEEE 24765c:2014) Note: includes both the demand for and the supply of IT services by internal groups, specialist IT units, or external suppliers and utility services (such as those providing software as services)

usefulness. (1) degree to which a user is satisfied with perceived achievement of pragmatic goals, including the results of use and the consequences of use (ISO/IEC 25010:2011 Systems and software engineering--System and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.1.3.1)

product during its utilization (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.26) (9) individual or organization that uses the system or software to perform a specific function (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.40) (10) individual who or group that benefits from a system during its utilization (INCOSE Systems Engineering Handbook, 4th ed.) (11) person who interacts with an autonomous/intelligent system (IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1) Note: The user can perform other roles, such as acquirer or maintainer. The role of user and the role of operator may be vested, simultaneously or sequentially, in the same individual or organization. See also: developer, end user, functional user, indirect user, operator, secondary user

**user documentation.** (1) documentation for users of a system, including a system description and procedures for using the system to obtain desired results (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) information to describe, explain, or instruct how to use software (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.55) (3) information that is supplied with the software to help the users in their use of that software (ISO/IEC 25051:2014 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.26) See also: user manual

**user error protection.** (1) degree to which a system protects users against making errors (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.4.4) (2) degree to which an IT service protects users against making errors (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.2.4)

**user experience.** (1) person's perceptions and responses that result from the use or anticipated use of a product, system or service (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.20) Note: User experience is a consequence of brand image, presentation, functionality, system performance, interactive behavior, and assistive capabilities of the interactive system; the user's internal and physical state resulting from prior experiences, attitudes, skills and personality; and the context of use.

**user group.** (1) subset of intended users who are differentiated from other intended users by factors such as age, culture or expertise that are likely to influence usability (ISO/IEC 25062:2006 Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Common Industry Format (CIF) for usability test reports, 4.7)

**user interaction.** (1) exchange of information between a user and an interactive system via the user interface to complete the intended task (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.22) Note: User interaction specifications focus on user interactions without considering implementation details.

**user interface.** (1) components of an interactive system (software or hardware) that provide information and controls
for the user to accomplish specific tasks with the interactive system (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.23) (2) ensemble of software and hardware that allows a user to interact with a system (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.56) (ISO/IEC/IEEE 26512:2018 Systems and software engineering--Requirements for acquirers and suppliers of information for users, 3.29) (3) interface that enables information to be passed between a human user and hardware or software components of a computer system (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (4) all components of an interactive system (software or hardware) that provide information and controls for the user to accomplish specific tasks with the interactive system (ISO/IEC 25063:2014 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description)

user interface aesthetics. (1) degree to which a user interface enables pleasing and satisfying interaction for the user (ISO/IEC 25010:2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models, 4.2.4.5) Note: refers to properties of the product or system that increase the pleasure and satisfaction of the user, such as the use of color and the nature of the graphical design

user interface element. (1) entity of the user interface that is presented to the user by the software (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.24) Note: User interface elements can be interactive or not, and either entities relevant to the task or entities of the user interface

Syn: user interface object

user interface task. (1) task that hides the details of the interface to and interacts sequentially with a human user (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

user manual. (1) document that presents the information necessary to employ a system or component to obtain desired results (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) document that describes how to use a functional unit, and that can include description of the rights and responsibilities of the user, the owner, and the supplier of the unit (ISO/IEC 2382:2015 Information technology -- Vocabulary) Note: Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose. Syn: user guide, user's manual, users' manual See also: data input sheet, diagnostic manual, installation manual, maintenance manual, operator manual, programmer manual, support manual

user need. (1) prerequisite identified as necessary for a user, or a set of users, to achieve an intended outcome, implied or stated within a specific context of use (ISO/IEC 25064:2013 Systems and software engineering--Software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: User needs report, 4.19) (2) set of functional user requirements and non-functional user requirements that the users need the system to fulfill (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FiSMA 1.1 functional size
user profile. (1) set of attributes that are unique to a specific user or user group, such as job function or subscription to a service, used to control the parts of the system or web page that users can access. (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.26)

user requirements (UR). (1) requirements for use that provide the basis for design and evaluation of interactive systems to meet identified user needs. (ISO/IEC TR 25060:2010 Systems and software engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for usability: General framework for usability-related information, 2.21) (2) description of the set of user needs for the software. (ISO/IEC 14143-1:2007 Information technology--Software measurement--Functional size measurement; Part 1: Definition of concepts, 3.12) (3) expression of perceived need from individual or group that benefits from a system during its utilization. (ISO/IEC TR 24766:2009 Information technology--Systems and software engineering--Guide for requirements engineering tool capabilities, 3.10) Note: User requirements specify the extent to which user needs and capabilities are to be met when using the system. They are not requirements on the users. User requirements are derived from user needs and capabilities in order to make use of the system in an effective, efficient, safe and satisfying manner. User requirements comprise two subsets: functional user requirements and non-functional user requirements.

[ISO 25063:2014] In software-engineering terms, user requirements comprise both "functional" and "non-functional" requirements based on user needs and capabilities. Syn: usage requirements

user story. (1) simple narrative illustrating a user requirement from the perspective of a persona. (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 3.16) (2) a narrative description of a software requirement, function, feature, or quality attribute, presented as a narrative of desired user interactions with a software system. (Software Extension to the PMBOK(R) Guide Fifth Edition)

user terminal. (1) terminal that enables a user to communicate with a computer. (ISO/IEC 2382:2015 Information technology --Vocabulary)

user type. (1) classification of emulated users that is defined by the combination of 1) the relative frequencies of the use of chain types; 2) the preparation times (mean values and their standard deviations). (ISO/IEC 14756:1999 Information technology -- Measurement and rating of performance of computer-based software systems, 4.32)

user view. (1) Functional User Requirements as perceived by the user. (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.52) (2) the application as seen through the eyes of the user. (ISO/IEC 24570:2018 Software engineering -- NESMA functional size measurement method -- Definitions and counting guidelines for the application of function point analysis) Note: Developers translate the user view into software to provide a solution. Syn: user perspective

user-based evaluation. (1) evaluation that involves representative users performing tasks with the system to enable identification of usability problems or measurements of efficiency, effectiveness, user satisfaction, or other user experiences. (ISO/IEC 25066:2016, Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--Common Industry Format (CIF) for Usability--Evaluation Report, 3.22)

user-friendly. (1) pertaining to ease and convenience of use by humans. (ISO/IEC 2382:2015 Information technology -
- Vocabulary (2) pertaining to a computer system, device, program, or document designed with ease of use as a primary objective (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

user-recognizable. (1) of requirements for processes or data, agreed upon and understood by both the user and the software developer (ISO/IEC 20926:2009 Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009, 3.51) Syn: user recognizable

UTC. (1) Coordinated Universal Time (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

utility. (1) software tool designed to perform some frequently used support function (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) measure of value within a given value system, often measured on a scale of 0 to 100 (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

utilization. (1) in computer performance evaluation, a ratio representing the amount of time a system or component is busy divided by the time it is available (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: busy time, idle time, up time

utilization bound theorem. (1) real-time scheduling theorem stating the conditions under which a set of n independent periodic tasks scheduled by the rate-monotonic algorithm will always meet their deadlines (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

V&V. (1) verification and validation (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2) Syn: V & V

VAC. (1) variance at completion (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

valid. (1) status of an information structure that follows the specified XML Schema document and is valid from an XML perspective (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.52) (2) status of a resource utilization measure (RUM) that follows the specified XML Schema document and is valid from an XML perspective (ISO/IEC 19770-4:2017 Information technology -- IT asset management -- Part 4: Resource utilization measurement, 3.11)

validate scope. (1) the process of formalizing acceptance of the completed project deliverables (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

validation. (1) confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.41) (ISO/IEC/IEEE 12207:2017 Systems and software engineering--Software life cycle processes, 3.1.71) (ISO/IEC/IEEE 15288:2015 Systems and software engineering--System life cycle processes, 4.1.53) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.61) (2) process of providing evidence that the system, software, or hardware and its associated products satisfy requirements allocated to it at the end of each life cycle activity, solve the right problem (e.g., correctly model physical laws, implement business rules, and use the proper system assumptions), and satisfy intended use and user needs (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.35) (3) the assurance that a product, service, or system meets the needs of the
customer and other identified stakeholders (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) - Sixth Edition) process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) confirmation in a timely manner, through automated techniques where possible, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: Validation in a system life cycle context is the set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals, and objectives (meet stakeholder requirements) in the intended operational environment. The right system has been built or is operating to meet business objectives. Validation demonstrates that the system can be used by the users for their specific tasks. "Validated" is used to designate the corresponding status. [ISO 9000:2005] Multiple validations can be carried out if there are different intended uses. See also: verification

validation action. (1) action that describes what is to be validated (the element as reference), on which item the action is performed, the expected result from the performance of the action, the validation technique to apply, and at which level of decomposition of the system-of-interest (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.15)

validation data. (1) dataset used to evaluate a candidate machine learning model while tuning it (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.86)

validation test. (1) test to determine whether an implemented system fulfils its specified requirements (ISO/IEC 2382:2015 Information technology -- Vocabulary)

value. (1) number or category assigned to an attribute of an entity by making a measurement (ISO/IEC 25000:2014 Systems and software Engineering--Systems and software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, 4.42) (2) numerical or categorical result assigned to a base measure, derived measure, or indicator (ISO/IEC/IEEE 15939:2017 Systems and software engineering--Measurement process, 3.41) (3) entity that is possibly an actual parameter in a request (ISO/IEC 19500-2:2012 Information technology --Object Management Group--Common Object Request Broker Architecture (CORBA)--Part 2: Interoperability, 3.2.26) (4) regard that something is held to deserve; the importance, worth, or usefulness of something to somebody (ISO/IEC/IEEE 42030:2019 Software, systems, and enterprise--Architecture evaluation framework, 3.10) (5) measure of worth (e.g., benefit divided by cost) of a specific product or service by a customer, and potentially other stakeholders and is a function of (i) the product's usefulness in satisfying a customer need, (ii) the relative importance of the need being satisfied, (iii) the availability of the product relative to when it is needed, and (iv) the cost of ownership to the customer (INCOSE Systems Engineering Handbook, 4th ed.) Note: Value can be tangible or intangible, financial or non-financial, and includes consideration of risks and liabilities. Realization of value will normally involve a balancing of costs, risks, opportunities and performance benefits. These words have similar but not identical meaning. Worth is usually what one is willing to pay for something. Significance is about being worthy of attention. Importance is about the state or fact of being of great significance or value. Usefulness is about serving some
purpose, or about being advantageous, helpful or of good effect. Benefit is about an advantage or profit gained from something. Quality is about the degree of excellence of something.

**value baseline.** (1) measure of a set of assets before an optimization, assigning relevant values to each group of assets being tracked (ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.53)

**value chain analysis.** (1) entire sequence of activities or parties that provide or receive value in the form of products or services (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.41)

**value change coverage.** (1) proportion of neurons activated where their activation values differ by more than a change amount divided by the total number of neurons in the neural network (normally expressed as a percentage) for a set of tests (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.87)

**value class.** (1) class that represents instances that are pure values (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.202) Note: The constituent instances of a value class do not come and go and cannot change state.

**value engineering (VE).** (1) analysis of features, systems, infrastructure, and material selections to realize essential functions at the lowest life cycle cost consistent with required performance, quality, reliability, and safety (ISO/IEC/IEEE 24765h:2019)

**value list constraint.** (1) constraint that specifies the set of all acceptable instance values for a value class (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.203)

**value range constraint.** (1) constraint that specifies the set of all acceptable instance values for a value class where the instance values are constrained by a lower and/or upper boundary (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.204) Note: A range constraint only makes sense if there is a linear ordering specified.

**value-added reseller (VAR).** (1) company licensed to repackage and support existing products, such as combined software packages (ISO/IEC/IEEE 24765g:2018) Syn: value added reseller


**variability.** (1) characteristics that can differ among members of the product line (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.23) (2) set of functional and non-functional characteristics that can differ among members of the product line (ISO/IEC 26560:2019 Software and systems engineering -- Tools and methods for product line product management, 3.7)

**variability binding.** (1) act of determining the variant of the variation point defined in the variability model (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.7) (2) selection of a certain variant for a variation point (ISO/IEC 26554:2018 Information technology--Software and systems engineering--Tools and methods for product line testing, 3.10)
variability constraint. (1) constraint relationships between a variant and a variation point, between two variants, and between two variation points (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.24)

variability dependency. (1) relationship between a variation point and a set of variants, which indicates that the variation point implies a decision about the variants (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.25) (2) association from a variation point to a variant or variants (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.7)

variability documentation. (1) detailed description of variability models being used across the member products within a product line (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.8)

variability implementation. (1) variability development in source codes or executable modules (ISO/IEC 26553:2018 Information technology-Software and systems engineering- Tools and methods for product line realization, 3.18)

variability in requirements. (1) external and internal variability in requirements engineering (ISO/IEC 26551:2016 Software and systems engineering --Tools and methods for product line requirements engineering, 3.20)

variability in space. (1) variation that occurs at the same time with a different shape (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.9)

variability in test cases. (1) variability included in domain test cases that will be bound during application testing in order to derive concrete test cases (ISO/IEC 26554:2018 Information technology--Software and systems engineering--Tools and methods for product line testing, 3.11)

variability in time. (1) variation that occurs at different times (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.10)

variability management. (1) managerial tasks relate to variability and has two dimensions: variability dimension and asset dimension (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.26)


variability mechanism operationalization (VMO). (1) adequate provision or binding of variability mechanisms at each specific domain or application engineering life cycle stage (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.15)

variability model. (1) explicit definition for product line variability (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.27) Note: It introduces variation points (3.19), types of variation for the variation points, variants offered by the variation points, variability dependencies and
variability constraints. Variability models may be orthogonal to or integrated in other models such as requirements or design models. There are two types of variability models: application variability models and domain variability models.

**variability modeling plan.** (1) documentation that includes schedules, defined roles and responsibilities, and defined quality assurance measures that will be applied to variability modelling (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.9) Syn: variability modelling plan

**variability modeling strategy.** (1) variability modelling methodology, strictness degree of variability model validation, rules, constraints, other details for supporting the role of variability model in the whole variability management (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.10) Syn: variability modelling strategy

**variability traceability.** (1) trace links established for a variability model with both domain assets and application assets where variants are bound (ISO/IEC 26555:2015 Software and systems engineering--Tools and methods for product line technical management, 3.12) (2) traceability among variability definition, variability implementation in domain engineering and application engineering, and thereafter tracing impacts due to variability changes (ISO/IEC 26559:2017 Software and systems engineering -- Methods and tools for variability traceability in software and systems product line, 3.8)


**variable cost.** (1) cost, such as the cost of material, which is directly dependent on the rate of production (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: fixed cost

**variable definition.** (1) see data definition (ISO/IEC/IEEE 29119-4:2015 Software and systems engineering -- Software testing -- Part 4: Test techniques, 4.28)

**variable trace.** (1) record of the name and values of variables accessed or changed during the execution of a computer program (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: data-flow trace, data trace, value trace See also: execution trace, retrospective trace, subroutine trace, symbolic trace

**variance.** (1) a quantifiable deviation, departure, or divergence away from a known baseline or expected value. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: variation

**variance analysis.** (1) a technique for determining the cause and degree of difference between the baseline and actual performance (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**variance at completion (VAC).** (1) a projection of the amount of budget deficit or surplus, expressed as the difference between the budget at completion and the estimate at completion (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

**variant.** (1) fault tolerance, a version of a program resulting from the application of software diversity (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) one alternative that is used to realize particular variation
variant selection. (1) decision making for a choice of a variant in a variation point (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.17) Note: One or more variants should correspond to each variation point to represent V_VP relationship. Selection and binding of variants for a specific product determine the characteristics of the particular variability for the product.

variation. (1) an actual condition that is different from the expected condition that is contained in the baseline plan (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) See also: variance

variation point. (1) representation corresponding to particular variable characteristics of products, domain assets, and application assets in the context of a product line (ISO/IEC 26550:2015 Software and systems engineering--Reference model for product line engineering and management, 3.29) (2) indication of product differentiation based on particular variable characteristics of products, domain assets, and application assets in the context of a product line (ISO/IEC 26558:2017 Software and systems engineering -- Methods and tools for variability modelling in software and systems product line, 3.12) Note: In principle, each variation point has at least one variant.

VCRM. (1) verification cross-reference matrix (IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2) See also: RTM

VDD. (1) version description document (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) velocity. (1) the rate of current work unit completion, measured as work units completed per fixed time period, such as story points, delivered features, functions, function points, user stories, use cases, or requirements completed in a given time period. Used as a measure of burndown rate or burnup rate. (Software Extension to the PMBOK(R) Guide Fifth Edition)

vendor branch. (1) branch for keeping track of versions of imported software (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) verb phrase. (1) part of the label of a relationship that names the relationship in a way that a sentence can be formed by combining the first class name, the verb phrase, the cardinality expression, and the second class name or role name (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.206) (2) phrase used to name a relationship, which consists of a verb and words that constitute the object of the phrase (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.206) Note: A verb phrase is ideally stated in active voice.

processes, 4.1.54) (ISO/IEC/IEEE 24748-1:2018 Systems and software engineering--Life cycle management--Part 1: Guidelines for life cycle management, 2.62) (2) evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (3) 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 (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.36) (4) process of providing objective evidence that the system, software, or hardware and its associated products conform to requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance), satisfy standards, practices, and conventions during life cycle processes, and successfully complete each life cycle activity and satisfy all the criteria for initiating succeeding life cycle activities (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1) (5) confirmation in a timely manner, using automated techniques where possible, through the provision of objective evidence, that specified requirements have been fulfilled (IEEE 2675-2021, IEEE Standard for DevOps: Building Reliable and Secure Systems Including Application Build, Package, and Deployment, 3.1) Note: Verification in a life cycle context is a set of activities that compares a product of the life cycle against the required characteristics for that product. This can include, but is not limited to, specified requirements, design description, and the system itself. The system has been built right. "Verified" is used to designate the corresponding status. Verification of interim work products is essential for proper understanding and assessment of the life cycle phase product(s). A system may be verified to meet the stated requirements, yet be unsuitable for operation by the actual users. See also: validation

verification action. (1) action that describes what is to be verified (the element as reference), on which item the action is performed, the expected result from the performance of the action, the verification technique to apply, and at which level of decomposition of the system-of-interest (ISO/IEC TS 24748-6:2016 Systems and software engineering--Life cycle management--Part 6: System integration engineering, 3.1.17)

verification and validation (V&V). (1) 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 (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: independent verification and validation

verification and validation (V&V) effort. (1) work associated with performing the V&V processes, activities, and tasks (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.1.37)

verification method. (1) method for producing objective evidence that specified requirements of a system have been fulfilled (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.34) (2) a method that tests an FSM method, and provides objective evidence of the extent to which a particular performance property is exhibited (ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.1) Note: Verification of an FSM method should produce a result that indicates the extent to which a performance property is exhibited, or whether a performance property is exhibited to a stated extent. For this reason, there is no concept of "pass" or "fail". An FSM method can be considered to be either "verified" or "not verified", for a particular performance property, based on whether
or not the appropriate verification has been conducted.

**verification sponsor.** (1) the person or organization that requires the verification to be performed and provides financial or other resources to carry it out *(ISO/IEC TR 14143-3:2003 Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 3.11)*

**verification test.** (1) test of a system to prove that it meets all its specified requirements at a particular stage of its development *(ISO/IEC 2382:2015 Information technology -- Vocabulary)*

**verification tool.** (1) instrument that can be used during verification to collect information about the target of verification, to interpret information, or to automate part of the verification *(ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.35)*

**verified deliverable.** (1) completed project deliverable that has been checked and confirmed for correctness through the control quality process *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**Verilog hardware description language (VHDL).** (1) hardware description language used to design and verify digital circuits *(ISO/IEC/IEEE 24765e:2015)* Note: standardized in IEEE 1394.

**version.** (1) initial release or re-release of a computer software configuration item, associated with a complete compilation or recompilation of the computer software configuration item *(IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)* (2) initial release or complete re-release of a document, as opposed to a revision resulting from issuing change pages to a previous release *(IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)* (3) operational software product that differs from similar products in terms of capability, environmental requirements, and configuration *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* (4) identified instance of a configuration item *(ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration management tool capabilities, 3.15)* (5) unique string of number and letter values indicating a unique revision of an item *(ISO/IEC 19770-5:2015 Information technology--IT asset management--Overview and vocabulary, 3.54)* Note: Versions often identify revisions of software that provide unique functionality or fixes. A version typically has multiple parts, such as a major version, indicating large changes in functionality or user interface changes, and a minor version, indicating smaller changes in functionality or user interface changes. See also: release

**version control.** (1) establishment and maintenance of baselines and the identification and control of changes to baselines that make it possible to return to the previous baseline *(ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.40)* See also: change control

**version description document (VDD).** (1) document that accompanies and identifies a given version of a system or component *(ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)* Note: Typical contents include an inventory of system or component parts, identification of changes incorporated into this version, and installation and operating information unique to the version described.

**version identifier.** (1) supplementary information used to distinguish a version of a configuration item from other versions *(ISO/IEC TR 18018:2010 Information technology--Systems and software engineering--Guide for configuration*
management tool capabilities, 3.16) Note: Version numbers are used to identify the version of the software product being compared with another version.

versioning. (1) assignment of either unique version names or unique version numbers to unique states of software configuration items, usually for a specific purpose, such as a release of the software product to an external group or the identification of a specific baseline (IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1)

vertical microinstruction. (1) microinstruction that specifies one of a sequence of operations needed to carry out a machine language instruction (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: Vertical microinstructions are relatively short, 12 to 24 bits, and are called ‘vertical’ because a sequence of such instruction, normally listed vertically on a page, is required to carry out a single machine language instruction. See also: diagonal microinstruction, horizontal microinstruction

very small entity (VSE). (1) entity (enterprise, organization, department or project) having up to 25 people (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 2.41)

VHSIC. (1) very high speed integrated circuit (ISO/IEC/IEEE 24765e:2015)

video display terminal (VDT). (1) user terminal with a display screen and usually equipped with an input unit such as a keyboard (ISO/IEC 2382:2015 Information technology -- Vocabulary) Syn: visual display terminal, visual display unit (VDU)

video random access memory (VRAM). (1) random access memory designed for the frame buffers of graphics cards (ISO/IEC/IEEE 24765e:2014)

view. (1) developer’s copy of a branch (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (2) collection of subject domains, classes, relationships, responsibilities, properties, constraints, and notes assembled or created for a certain purpose and covering a certain scope (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.207) (3) collection of entities and assigned attributes (domains) assembled for some purpose (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.207) (4) set of related categories (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) (5) representation of a whole system from the perspective of a related set of concerns (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model (KDM), 4) (6) &lt;architecture&gt; information item expressing the architecture from the perspective of specific stakeholders regarding specific aspects of the architecture entity and its environment (ISO/IEC/IEEE 42020:2019 Software, systems and enterprise -- Architecture processes, 3.24) Note: A view can cover the entire area being modeled or only a part of that area. When the term view is used without any qualifier it refers to the general case. When a qualifier is prepended to the word view, this indicates that the architecture view is specific to a particular viewpoint, e.g., operational view or services view.


viewpoint. (1) specification of the conventions for constructing and using a view (ISO/IEC 19506:2012 Information technology -- Object Management Group Architecture-Driven Modernization (ADM) -- Knowledge Discovery Meta-Model
viewpoint (on a system). (1) a form of abstraction achieved using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within a system (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.7)

viewpoint correspondence. (1) statement that some terms or other linguistic constructs in a specification from one viewpoint are associated with (e.g., describe the same entities as) terms or constructs in a specification from a second viewpoint (ISO/IEC 10746-2:2009 Information technology -- Open Distributed Processing -- Reference Model: Foundations, 3.2.8) Note: The forms of association that can be expressed will depend on the specification technique used. The terms associated by a correspondence need not necessarily be expressed using a single specification technique. The correspondence can associate a term in one specification technique with a term in some different specification technique. Rather than linking every individual pair of terms, general correspondences can also be expressed between specification techniques themselves. For example, composition operators defined in different specification techniques can be associated, implying correspondences wherever these operators are used to link terms in the respective viewpoints.

viewpoint statement. (1) brief statement of the perspective of an IDEF0 model that is presented in the a-0 context diagram of the model (IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.109)

violation. (1) behavior contrary to that required by a rule (ISO/IEC 15414:2015 Information technology -- Open distributed processing -- Reference model -- Enterprise language, 6.3.8) (2) behavior, act, or event deviating from a system's desired property or claim of interest (ISO/IEC/IEEE 15026-1:2019 Systems and software engineering--Systems and software assurance--Part 1: Concepts and vocabulary, 3.4.8) Note: In the area of safety, the term "violation" is used to refer to a deliberate human contravention of a procedure or rule. A rule or policy can provide behavior to occur upon violation of that or some other rule or policy.

virtual. (1) pertaining to a functional unit that appears to be real, but whose functions are accomplished by other means (ISO/IEC 2382:2015 Information technology -- Vocabulary) (2) for an entity, being composed of one or more underlying base entities (ISO/IEC 15476-4:2005 Information technology--CDIF semantic metamodel--Part 4: Data models, 6.16)

virtual address. (1) in a virtual storage system, the address assigned to an auxiliary storage location to allow that location to be accessed as though it were part of main storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) See also: real address

virtual machine (VM). (1) pertaining to a functional unit that appears to be real, but whose functions are accomplished by other means (ISO/IEC 2382:2015 Information technology -- Vocabulary)
virtual reference. (1) references made to concepts other than specific meta-entities in a metamodel (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: represented by boxes with diagonal striping.

virtual storage. (1) storage allocation technique in which auxiliary storage can be addressed as though it were part of main storage (ISO/IEC/IEEE 24765:2017 Systems and software engineering--Vocabulary) Note: Portions of a user's program and data are placed in auxiliary storage, and the operating system automatically swaps them in and out of main storage as needed. Syn: multilevel storage, virtual memory See also: real storage, virtual address, paging

virtual system of systems. (1) system of systems (SoS) that lacks a central management authority and a centrally-agreed-upon purpose for the SoS (ISO/IEC/IEEE 21841:2019 Systems and software engineering--Taxonomy of systems of systems, 3.2.4) Note: Virtual SoS are typically self-organizing. Large-scale behavior emerges and can be desirable, but this type of SoS relies on relatively invisible mechanisms to maintain it. Syn: virtual SoS

virtual team. (1) groups of people with a shared goal who fulfill their roles with little or no time spent meeting face to face (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

virtual test environment. (1) test environment where one or more parts are digitally simulated (ISO/IEC TR 29119-11:2020, Software and systems engineering--Software testing--Part 11: Guidelines on the testing of AI-based systems, 3.1.88)

visibility. (1) degree to which a transaction can access object state concurrently with other transactions (ISO/IEC 10746-3:2009 Information technology -- Open Distributed Processing -- Reference Model: Architecture, 13.7.1.3) (2) specification, for a property, of who can see it?&quot; (IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.209) (3) degree to which users have insight into the capabilities of the IT service, how services will be delivered, and progress toward their completion during delivery (ISO/IEC TS 25011:2017 Information technology--Systems and software Quality Requirements and Evaluation (SQuaRE)--Service quality models, 3.2.5.1) Note: Visibility is private, protected, or public.

vital business service. (1) service that is critical to the success of the business (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 3.39)

VMO. (1) variability mechanism operationalization (ISO/IEC 26557:2016 Software and systems engineering -- Methods and tools for variability mechanisms in software and systems product line, 3.15)

vocabulary. (1) collection of information related to a specific subset of terms related to a specific domain (ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.38) Note: Vocabulary is generally used to keep consistency, to avoid duplication, and to support synonyms.

voice of the customer. (1) a planning technique used to provide products, services, and results that truly reflect customer requirements by translating those customer requirements into the appropriate technical requirements for each phase of project product development. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

volatile memory. (1) unit that stores data only while power is on (ISO/IEC/IEEE 24765c:2014) See also: non-volatile
memory volume testing. (1) type of performance efficiency testing conducted to evaluate the capability of the test item to process specified volumes of data (usually at or near maximum specified capacity) in terms of throughput capacity, storage capacity, or both (ISO/IEC/IEEE 29119-1:2013 Software and systems engineering--Software testing--Part 1: Concepts and definitions, 4.95)

VRAM. (1) video random access memory (ISO/IEC/IEEE 24765c:2014)


VSE. (1) very small entity (ISO/IEC TR 29110-1:2016 Software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 1: Overview, 3.2) (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 4.2) Note: The plural is VSEs (very small entities)

vulnerability. (1) potential flaw or weakness in software design or implementation that could be exercised (accidentally triggered or intentionally exploited) and result in harm to the system (ISO/IEC 23643:2020, Software and systems engineering--Capabilities of software safety and security verification tools, 3.36)

VVP. (1) verification and validation plan (IEEE 1012-2016 IEEE Standard for System, Software, and Hardware Verification and Validation, 3.2)


waiver. (1) written authorization to accept a configuration item or other designated item which, during production or after having been submitted for inspection, is found to depart from specified requirements, but is nevertheless considered suitable for use as is or after rework by an approved method (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: configuration control, deviation, engineering change

walk-through. (1) static analysis technique in which a designer or programmer leads members of the development team and other interested parties through a segment of documentation or code, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) formal review in which an author leads members of the review through a work product, and the participants ask questions and make comments about possible issues (ISO/IEC 20246:2017 Software and systems engineering -- Work product reviews, 3.19) Syn: walkthrough


war room. (1) room used for project conferences and planning, often displaying charts of cost, schedule status, and other key project data. (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

warning. (1) advisory information in documentation that states that performing some action can lead to serious or dangerous consequences (ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.57) (2) hazardous situation, which if not avoided, can result in death or serious injury (ISO/IEC/IEEE 26513:2017 Systems and software engineering--Requirements for testers and reviewers of
information for users, 3.50) See also: caution, note

warning message. (1) safety-related information that warns about hazards and instructs on how to avoid them (IEC/IEEE 82079-1:2019 Preparation of information for use (instructions for use) of products: Part 1: Principles and general requirements, 3.48) Note: Warning messages are normally given within step-by-step instructions related to hazardous tasks.

waste. (1) work that adds no value to the product or service in the eyes of the customer (INCOSE Systems Engineering Handbook, 4th ed.)

watchdog timer (WDT). (1) electronic unit that triggers a reset of an embedded system or other corrective action if the main program, due to some fault condition, fails to periodically signal it. (ISO/IEC/IEEE 24765d:2015)

waterfall model. (1) model of the software development process in which the constituent activities, typically a concept phase, requirements phase, design phase, implementation phase, test phase, and installation and checkout phase, are performed in that order, possibly with overlap but with little or no iteration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: incremental development, rapid prototyping, spiral model

WAV. (1) Waveform audio file format (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 5)


WCAG. (1) Web Content Accessibility Guidelines (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 5)

WDT. (1) watchdog timer (ISO/IEC/IEEE 24765d:2015)

wearout-failure period. (1) period in the life cycle of a system or component during which hardware failures occur at an increasing rate due to deterioration (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: constant-failure period, early-failure period, bathtub curve

Web Content Accessibility Guidelines (WCAG). (1) set of guidelines that specify how to make content accessible, primarily for people with disabilities, but also for all user agents (IEEE 2430-2019 Trial-Use Standard for Software Non-Functional Sizing Measurements, 3.1)

web page. (1) coherent presentation of a set of content objects and associated interaction objects delivered to users through a browser in accordance with Internet protocols (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.28) Note: A web page can be generated dynamically from the server side, and can incorporate multimedia, applets, or other elements active on either the client or server side.

webmaster. (This term is deprecated.) (1) person or group responsible to the website owner for ongoing maintenance of the site's presentation and availability (ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.27) Syn: website administrator, website manager

---

This definition is copyrighted ©, 2021 by the IEEE. The reader is granted permission to copy the definition as long as the statement “Copyright ©, 2021, IEEE. Used by permission.” remains with the definition. All other rights are reserved.

Copyright © 2021 ISO/IEC.

In accordance with ISO/IEC JTC 1/SC 7 N2882 and N2930, this definition is made publicly available. Permission is granted to copy the definition providing that its source is cited.


PMI is a service and trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.
**Website.** (1) Collection of logically connected web pages managed as a single entity *(ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.29)* Syn.: web site

**Website owner.** (1) Organization responsible for the site content and site design *(ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.30)* Note: The website owner can select a supplier as the website provider or can also be the website provider.

**Website provider.** (1) Organization responsible for operation of the website and delivery of site content to users *(ISO/IEC/IEEE 23026:2015 Systems and software engineering--Engineering and management of websites for systems, software, and services information, 4.31)* Note: The website provider can also be the site owner, webmaster, site designer, or the internet service provider for the site.

**Weighted milestone method.** (1) Earned value method that divides a work package into measurable segments, each ending with an observable milestone, and assigns a weighted value to the achievement of each milestone *(ISO/IEC/IEEE 24765h:2019)*

**Well-being.** (1) Continuous and sustainable physical, mental, and social flourishing of individuals, communities and populations where their economic needs are cared for within a thriving ecological environment *(IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)*

**Well-being metrics.** (1) Subjective and objective indicators measuring both internal phenomena and external factors, encompassing the capabilities and subjective well-being approaches, and including but not limited to the domains of a) affect, b) community, c) culture, d) education, e) economy, f) environment, g) human settlements, h) health, i) government, j) psychological well-being/mental well-being, k) satisfaction with life, and l) work *(IEEE 7010-2020, IEEE Recommended Practice for Assessing the Impact of Autonomous and Intelligent Systems on Human Well-Being, 2.1)*

**What-if scenario analysis.** (1) The process of evaluating scenarios in order to predict their effect on project objectives *(A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)*

**Window.** (1) Area with visible boundaries that presents a view of a software object or through which a user conducts a dialog with a computer system *(ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.58)*

**Wizard.** (1) Procedural form of help that guides a user through each step of a task through dialog with the user *(ISO/IEC 26514:2008 Systems and software engineering--requirements for designers and developers of user documentation, 4.59)*

**Word.** (1) Sequence of bits or characters that is stored, addressed, transmitted, and operated on as a unit within a given
computer (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) element of computer storage that can hold a sequence of bits or characters as in (1) (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (3) sequence of bits or characters that has meaning and is considered an entity in some language (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: computer word

work authorization. (1) permission and direction, typically written, to begin work on a specific schedule activity or work package or control account (ISO/IEC/IEEE 24765h:2019) Note: It is a method for sanctioning project work to ensure that the work is done by the identified organization, at the right time, and in the proper sequence.

work breakdown structure (WBS). (1) a hierarchical decomposition of the total scope of work to be carried out by the project team to accomplish the project objectives and create the required deliverables (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) (2) deliverable-oriented hierarchical decomposition of the work to be executed by the project team to accomplish the project objectives and create the required deliverables (ISO/IEC/IEEE 26511:2018 Systems and software engineering--Requirements for managers of information for users of systems, software, and services, 3.1.42) (ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-6-2: Systems engineering--Management and engineering guide: Generic profile group: Basic profile, 3.12) Note: The WBS organizes and defines the total scope of the project.

work breakdown structure component. (1) an entry in the work breakdown structure that can be at any level. (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)


work effort. (1) labor resources required for the production of a specified output (ISO/IEC/IEEE 24765a:2011) Note: Labor resources are usually expressed as work hours.

work package. (1) the work defined at the lowest level of the work breakdown structure for which cost and duration can be estimated and managed (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

work performance data. (1) the raw observations and measurements identified during activities being performed to carry out the project work (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

work performance information. (1) the performance data collected from various controlling processes, analyzed in context and integrated based on relationships across areas (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition)

work performance reports. (1) the physical or electronic representation of work performance information compiled in project documents, intended to generate decisions, actions, or awareness (A Guide to the Project Management Body of Knowledge (PMBOK(R) Guide) -- Sixth Edition) Syn: performance reporting, performance reports

engineering—Lifecycle profiles for Very Small Entities (VSEs)—Part 1: Overview, 3.77) Note: Some subset of the work products will be baselined and some will form the set of project deliverables. Syn: WP

work unit. (1) a project task such as constructing or testing a function point, user story, feature, or requirement (Software Extension to the PMBOK(R) Guide Fifth Edition)

workaround. (1) method to continue operations despite a known error (ISO/IEC/IEEE 24765h:2019)

workflow board. (1) in software development, a visual representation of work for developers who pull tasks from the task backlog; used for on-demand or resource-bound scheduling (Software Extension to the PMBOK(R) Guide Fifth Edition) Syn: kanban board

working metamodel. (1) definition of the specific meta-objects that can be instantiated in the model section of a CDIF transfer (ISO/IEC 15474-1:2002 Information technology -- CDIF framework -- Part 1: Overview, 4.2) Note: The working metamodel comprises the meta-objects in the CDIF semantic metamodel that are used by the subject areas referenced in the metamodel section of the transfer, and the meta-objects defined as extensions in the metamodel section.

working set. (1) in the paging method of storage allocation, the set of pages that are most likely to be resident in main storage at any given point of a program's execution (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary)

working space. (1) that portion of main storage that is assigned to a computer program for temporary storage of data (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Syn: working area, working storage

workload. (1) mix of tasks typically run on a given computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) Note: Major characteristics include input/output requirements, amount and kinds of computation, and computer resources required.

workload model. (1) model used in computer performance evaluation, depicting resource utilization and performance measures for anticipated or actual workloads in a computer system (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: system model

workstation. (1) functional unit that usually has special purpose computing capabilities and includes user-oriented input units and output units (ISO/IEC 2382:2015 Information technology -- Vocabulary)

WP. (1) work product (ISO/IEC TR 29110-5-3:2018 Systems and software engineering--Lifecycle profiles for Very Small Entities (VSEs)--Part 5-3: Service delivery guidelines, 4.1)

write. (1) to record data in a storage device or on a data medium (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) (2) data movement that moves a data group lying inside the functional process to persistent storage (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 2.28) Syn: write type See also: read

write (-type). (1) a data movement type that moves a data group lying inside the functional process to persistent storage (ISO/IEC 19761:2011 Software engineering -- COSMIC: a functional size measurement method, 3.26) Note: A Write is considered to include certain associated data manipulations necessary to achieve the Write.

writer. (1) person designing or developing user documentation (ISO/IEC/IEEE 26515: 2018 Systems and software engineering: Developing information for users in an agile environment, 4.17) Syn: author
writing reference. (1) data storage entity or other record, or interface record to another software or system to which data is written in a BFC (ISO/IEC 29881:2010 Information technology--Software and systems engineering--FiSMA 1.1 functional size measurement method, 3.10) Note: The number of writing references is greater than 0 with all BFC types where it is applicable.

XFN. (1) X/Open Federated Naming (ISO/IEC/IEEE 24765i:2020)

XHTML. (1) Extended HyperText Markup Language (ISO/IEC/IEEE 24765i:2020)

XLIFF. (1) XML Localization Interchange File Format (ISO/IEC/IEEE 26531:2015 Systems and software engineering -- Content management for product lifecycle, user and service management documentation, 5)


XML. (1) extensible markup language (ISO/IEC 15909-2:2011 Software and system engineering -- High-level Petri nets -- Part 2: Transfer format, 4.2.9)


yesterday's weather. (1) a report of work performance in the most recent reporting period (Software Extension to the PMBOK(R) Guide Fifth Edition)

zero-address instruction. (1) computer instruction that contains no address fields (ISO/IEC/IEEE 24765:2017 Systems and software engineering-Vocabulary) See also: one-address instruction, two-address instruction, three-address instruction, four-address instruction