

AS/NZS ISO/IEC/IEEE 24765:2024
ISO/IEC/IEEE 24765:2017



Australian/New Zealand Standard™

Systems and software engineering — Vocabulary



AS/NZS ISO/IEC/IEEE 24765:2024

This Joint Australian/New Zealand Standard™ was prepared by Joint Technical Committee IT-015, Software and Systems Engineering. It was approved on behalf of Standards Australia's Standards Development and Accreditation Committee on 12 June 2024 and by the New Zealand Standards Approval Board on 5 June 2024.

This Standard was published on 28 June 2024.

The following are represented on Committee IT-015:

Australian Computer Society
Australian Digital Health Agency
Australian Society for Technical Communication (NSW)
Business NSW
Department of Defence (Australian Government)
Engineers Australia
Griffith University
IT Service Management Forum Australia
Institute of IT Professionals New Zealand
International Council on Systems Engineering (INCOSE) New Zealand
Systems Engineering Society of Australia
Universities New Zealand
University of New South Wales
University of Southern Queensland
University of Technology Sydney

This Standard was issued in draft form for comment as DR AS/NZS ISO/IEC/IEEE 24765:2024.

Keeping Standards up-to-date

Ensure you have the latest versions of our publications and keep up-to-date about Amendments, Rulings, Withdrawals, and new projects by visiting:

www.standards.org.au

www.standards.govt.nz

ISBN 978 1 76139 710 3

Australian/New Zealand Standard™

Systems and software engineering — Vocabulary

First published as AS/NZS ISO/IEC/IEEE 24765:2024.

COPYRIGHT

© ISO/IEC/IEEE 2024 — All rights reserved

© Standards Australia Limited/the Crown in right of New Zealand, administered by the New Zealand Standards Executive 2024

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher, unless otherwise permitted under the Copyright Act 1968 (Cth) or the Copyright Act 1994 (New Zealand).

Preface

This Standard was prepared by the Joint Standards Australia/Standards New Zealand Committee IT-015, Software and Systems Engineering.

The objective of this document is to provide a common vocabulary applicable to all systems and software engineering work falling within the scope of ISO/IEC JTC 1/SC 7, Software and systems engineering, and the IEEE Computer Society Systems and Software Engineering Standards Committee (IEEE-CS S2ESC).

This document is identical with, and has been reproduced from, ISO/IEC IEEE 24765:2017, *Systems and software engineering — Vocabulary*.

As this document has been reproduced from an international document, a full point substitutes for a comma when referring to a decimal marker.

Australian or Australian/New Zealand Standards that are identical adoptions of international normative references may be used interchangeably. Refer to the online catalogue for information on specific Standards.

The terms “normative” and “informative” are used in Standards to define the application of the appendices or annexes to which they apply. A “normative” appendix or annex is an integral part of a Standard, whereas an “informative” appendix or annex is only for information and guidance.

Contents

Preface	ii
Foreword	iv
Introduction	vi
1 Scope	1
1.1 General	1
1.2 Relationship of the print and internet-accessible versions	1
1.3 Vocabulary structure	1
1.4 PMI Glossary provisions	2
2 Normative references	2
3 Terms, definitions, and abbreviated terms	2
Annex A (informative) List of References	547
Bibliography	548

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is called to the possibility that implementation of this standard may require the use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. ISO/IEEE is not responsible for identifying essential patents or patent claims for which a license may be required, for conducting inquiries into the legal validity or scope of patents or patent claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance or a Patent Statement and Licensing Declaration Form, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from ISO or the IEEE Standards Association. Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

Use of IEEE Standards documents is wholly voluntary. IEEE documents are made available for use subject to important notices and legal disclaimers (see <http://standards.ieee.org/IPR/disclaimers.html> for more information).

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, SC 7, *Software and systems engineering*, in cooperation with the IEEE Computer Society Systems and Software Engineering Standards Committee, under the Partner Standards Development Organization cooperation agreement between ISO and IEEE.

Certain material contained in ISO/IEC/IEEE 24765 is reproduced, with permission, from *A Guide to the Project Management Body of Knowledge (PMBOK®) Guide — Fifth Edition*, copyright 2013, Project Management Institute.

This second edition cancels and replaces the first edition (ISO/IEC/IEEE 24765:2010), and has been editorially revised. Revisions in terms and definitions published in this second edition have been previously approved through the vocabulary maintenance procedures of ISO/IEC JTC 1/SC7, in

cooperation with the IEEE Computer Society. These revisions have been made available through the online vocabulary database used for this standard, maintained by the ISO/IEC JTC 1/SC7/SWG 22 Vocabulary Validation Team in cooperation with the IEEE Computer Society at www.computer.org/sevocab

Introduction

The systems and software engineering disciplines are continuing to mature while information technology advances. New terms are being generated and new meanings are being adopted for existing terms. This document was prepared to collect and standardize terminology. Its purpose is to identify terms currently in use in the field and standard definitions for these terms. It is intended to serve as a useful reference for those in the Information Technology field, and to encourage the use of systems and software engineering standards prepared by ISO/IEC JTC 1 and liaison organizations IEEE Computer Society and Project Management Institute (PMI). It provides definitions that are rigorous, uncomplicated, and understandable by all concerned.

While it is useful to find the meaning of a term, no word stands in isolation. This document makes it possible to search for related concepts and to view how a term is used in definitions of other terms.

Every effort has been made to use definitions from established systems and software engineering standards of ISO JTC 1/SC 7 and its liaison organizations IEEE Computer Society and the PMI. When existing standards were found to be incomplete, unclear or inconsistent with other entries in the vocabulary, however, new, revised, or composite definitions have been developed. Some definitions have been recast in a system, rather than software, context.

The vocabulary is offered in both print and internet-accessible versions for ease of reference and to encourage use of the source standards for the vocabulary. The online vocabulary database used for this standard is maintained by the ISO/IEC JTC 1/SC7/SWG 22 Vocabulary Validation Team in cooperation with the IEEE Computer Society at www.computer.org/sevocab

Australian/New Zealand Standard

Systems and software engineering — Vocabulary

1 Scope

1.1 General

Consistent with ISO vocabulary standards, each technical committee is responsible for standard terminology in its area of specialization. This document provides a common vocabulary applicable to all systems and software engineering work falling within the scope of ISO/IEC JTC 1/SC 7, *Software and systems engineering*, and the IEEE Computer Society Systems and Software Engineering Standards Committee (IEEE-CS S2ESC).

The scope of each concept defined has been chosen to provide a definition that is suitable for general application. In those circumstances where a restricted application is concerned, a more specific definition might be needed.

Terms have been excluded if they were:

- considered to be parochial to one group or organization;
- company proprietary or trademarked;
- multi-word terms whose meaning could be inferred from the definitions of the component words; and
- terms whose meaning in the information technology (IT) field could be directly inferred from their common English dictionary meaning.

1.2 Relationship of the print and internet-accessible versions

The primary tool for maintaining this vocabulary is a database that is modified in a controlled fashion. Hosted by the IEEE Computer Society, the SEVOCAB (systems and software engineering vocabulary) database is publicly accessible at www.computer.org/sevocab ISO/IEC/IEEE 24765 is issued periodically as a formal, published document reflecting a "snapshot" of the database.

The copyright notice provided with the database permits users to copy definitions from the database as long as the source of the definition is cited. Permitting public use of the definitions in the database is intended to encourage the use of other ISO/IEC JTC 1 and IEEE systems and software engineering standards.

1.3 Vocabulary structure

Entries in the vocabulary are arranged alphabetically. Blanks precede all other characters in alphabetizing. Hyphens and slashes (- and /) follow all other characters in alphabetizing.

Preferred terms are shown in **bold**. Synonyms or admitted terms (terms with the same meaning as the preferred term), are listed under the preferred term in plain text, and can be located by searching.

Terms, definitions, and notes use spelling preferred in the US. The use of capital letters has been minimized and generally limited to proper names and acronyms. In some cases, the source standard uses another correct spelling (such as behaviour rather than behavior, on-line rather than online). Technical terms in English often change form from two words to a hyphenated word to a single word as they become more familiar, e.g., real time to real-time to realtime. Hence, other correct spellings and capitalization of the terms, according to a national standard, an authoritative general dictionary or accepted style guide, can be used with the definitions.

An entry can consist of a single word, such as "software"; a phrase or compound term, such as "test case"; or an abbreviated term, such as "CDR". Phrases are given in their natural order (test plan) rather

than in reversed order (plan, test). Abbreviated terms can be listed separately as well as in parentheses following the source term. Terms that are verbs are shown without the infinitive marker "to".

After each term, numbered definitions are listed in order of preference, or from the most general to the more specific usages. The different definitions can show the use of a term as a noun, verb and adjective.

This document includes references to the active source standards for each definition, so that the use of the term can be further explored. The sources of most of the definitions are ISO JTC 1/SC 7 or IEEE Computer Society standards and the PMI Glossary, Fifth Edition. Sources are listed in the Bibliography. Additional sources for definitions drawn from outside the scope of systems and software engineering are in Annex A, List of References. In some cases, the same definition can also be found in other active or withdrawn standards. No source is shown if the original source standard has been withdrawn or archived and the definition has been retained in this vocabulary.

Notes (comments), Examples, and Figures taken from the source standards have been included to clarify selected definitions.

Cross-references are used to show a term's relationship to other terms in the dictionary: *cf.* refers to related terms that are not synonyms.

1.4 PMI Glossary provisions

The Project Management Institute (PMI) Glossary definitions have been included without alteration in accordance with the copyright agreement. Some of these terms and definitions are not worded according to ISO/IEC or IEEE styles. Many of these definitions include explanatory material. For other terms and other definitions that have ISO/IEC and IEEE standards as their source, explanatory matter is shown in the Notes and Examples.

2 Normative references

There are no normative references in this document.

NOTE The definitions in this document are drawn from normative standards and informative guidance documents, including ISO/IEC Technical Reports (TR). Where terms have multiple definitions, users should consult the source standards for further information on appropriate usage within a specific context.

3 Terms, definitions, and abbreviated terms

For the purposes of this document, the following terms and definitions apply.

ISO, IEC and IEEE maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org>
- ISO Online browsing platform: available at <http://www.iso.org/obp>
- IEEE Standards Dictionary Online: available at <http://dictionary.ieee.org>

3.1

1GL

1. first-generation language

cf. machine language

3.2

2GL

1. second-generation language

cf. assembly language

3.3**3D**

1. three-dimensional [ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information]

3.4**3GL**

1. third-generation language
cf. high order language

3.5**4GL**

1. fourth-generation language

3.6**5GL**

1. fifth-generation language

3.7**<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 1 to entry: Thus, engineering language: definitions of concepts and rules for the specification of an ODP system from the engineering viewpoint.

3.8**<X> domain**

1. set of objects, each of which is related by a characterizing relationship <X> to a controlling object [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 10.3]

3.9**<X> federation**

1. community of <x> domains [ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 5.1.2]

3.10**<X> group**

1. set of objects with a particular characterizing relationship <X> [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 10.1]

3.11**<x> interceptor**

1. engineering object in a channel, placed at a boundary between <x> domains [ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.11]

Note 1 to entry: An <x> 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

3.12**<x> pattern**

1. abstract specification of a composition of objects that results in any instance of the composition having a given property, named by <X> [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.8]

3.13**A-0 context diagram**

1. the only context diagram that is 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 Semantics for IDEF0*]

3.14**A-profile**

1. Application profile [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.15**ABC**

1. activity-based costing

3.16**abend**

1. abnormal end

3.17**abnormal end (abend)**

1. termination of a process prior to completion
cf. abort, exception

3.18**abort**

1. to terminate a process prior to completion
cf. abend, exception

3.19**absolute address****explicit address****specific 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
cf. relative address, relocatable address, symbolic address, absolute assembler, absolute code, absolute instruction

3.20**absolute assembler**

1. assembler that produces absolute code
cf. relocating assembler

3.21**absolute code****specific code**

1. code in which all addresses are absolute addresses
cf. relocatable code

3.22**absolute instruction**

1. computer instruction in which all addresses are absolute addresses
cf. direct instruction, effective instruction, immediate instruction, indirect instruction

3.23**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

cf. relocating loader

3.24

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 1 to entry: 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.

3.25

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

3.26

abstract design

1. generic form that needs specialization (further design work) to produce concrete designs **2.** design aimed at producing designs

3.27

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 **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] cf. data abstraction

3.28

AC

1. actual cost [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.29

acceptability

1. exposure to loss (financial or otherwise) that an organization is willing to tolerate from a risk

Note 1 to entry: 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.

3.30

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]

3.31

acceptable

1. meeting stakeholder expectations that can be shown to be reasonable or merited

3.32

acceptance criteria

1. criteria that a system or component must satisfy in order to be accepted by a user, customer, or other authorized entity **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® Guide) — Fifth Edition]
cf. requirement, test criteria

3.33**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]*

cf. acceptance testing, validation test

3.34**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 **2.** formal testing conducted to enable a user, customer, or other authorized entity to determine whether to accept a system or component *[IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1]*

cf. acceptance test, validation test

3.35**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® Guide) — Fifth Edition]*

3.36**access**

1. to obtain the use of a resource *[ISO/IEC 2382:2015, Information technology — Vocabulary]*

3.37**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]*

3.38**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]*

3.39**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.

3.40**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]*

3.41

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 (SQuaRE) — 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 (SQuaRE) — 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 (SQuaRE) — 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]

Note 1 to entry: 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. Context of use includes direct use or use supported by assistive technologies.

[SOURCE: ISO 9241-171:2008]

3.42

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]

3.43

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]

3.44

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]

3.45

accuracy

1. qualitative assessment of correctness, or freedom from error **2.** quantitative measure of the magnitude of error **3.** Within the quality management system, accuracy is an assessment of correctness [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition] cf. precision

3.46**accuracy of measurement**

1. the 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 1 to entry: Accuracy is a qualitative concept. The term precision is not a synonym for "accuracy". 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.

[SOURCE: ISO/IEC Guide 99:2007 International vocabulary of metrology — Basic and general concepts and associated terms]

3.47**ACIA**

1. asynchronous communication interface adapter

3.48**ACID**

1. Atomicity Consistency Isolation Durability [ISO/IEC 10746-1:1998 *Information technology — Open Distributed Processing — Reference model: Overview*]

3.49**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]

3.50**acquire project team**

1. the process of confirming human resource availability and obtaining the team necessary to complete project activities [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.51**acquirer****owner****purchaser**

1. stakeholder that acquires or procures a product or service from a supplier [ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes*, 4.1; ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.1; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.1] **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 1 to entry: 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.

3.52**acquisition**

1. process of obtaining a system, product, or service [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 3.2; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.2] **2.** obtaining human and material resources necessary to perform project activities. Acquisition implies a cost of resources, and is not necessarily financial [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.53**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

3.54**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] **5.** process of transformation that operates upon data or other types of inputs to create data, produce outputs, or change the state or condition of the subject software [IEEE 1175.3-2004 *IEEE Standard for CASE Tool Interconnections — Reference Model for Specifying Software Behavior*] **6.** statement of causal and affective relationships in a behavior model linking particular stimulus interactions to particular response interactions and changes within a unit under a certain set of conditions on a unit's lifeline [IEEE 1175.4-2008 *IEEE Standard for CASE Tool Interconnections — Reference Model for Specifying System Behavior*, 3.1]

3.55**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]

3.56**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]

3.57**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]

3.58**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]

3.59**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]

3.60**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*]

3.61**active area**

1. (on-screen documentation) area that responds to user input

EXAMPLE a window, icon or text field

3.62**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*]

3.63**active interconnection**

1. physical interaction mechanism allowing the action of one thing to cause a change or to stimulate an action in another thing [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.1*]

3.64**active redundancy**

1. in fault tolerance, the use of redundant elements operating simultaneously to prevent, or permit recovery from, failures
cf. standby redundancy

3.65**active text**

1. text displayed on the screen that responds to user input

3.66**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

3.67**activity**

1. set of cohesive tasks of a process [ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes*; ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.3; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.3] **2.** a distinct, scheduled portion of work performed during the course of a project [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth 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 **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 **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 1 to entry: An activity normally has an expected duration, cost, and resource requirements. Activities are often subdivided into tasks.

3.68**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® Guide) — Fifth Edition*]

3.69**activity code**

1. one or more numerical or text values that identify characteristics of the work or in some way categorize the schedule activity that allows filtering and ordering of activities within reports [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.70**activity cost estimates**

1. the projected cost of the schedule activity that includes the cost for all resources required to perform and complete the activity, including all cost types and cost components [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.71**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® Guide) — Fifth Edition*]
cf. duration

3.72**activity duration estimate**

1. a quantitative assessment of the likely amount or outcome for the duration of an activity [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.73**activity group**

1. set of related activities.

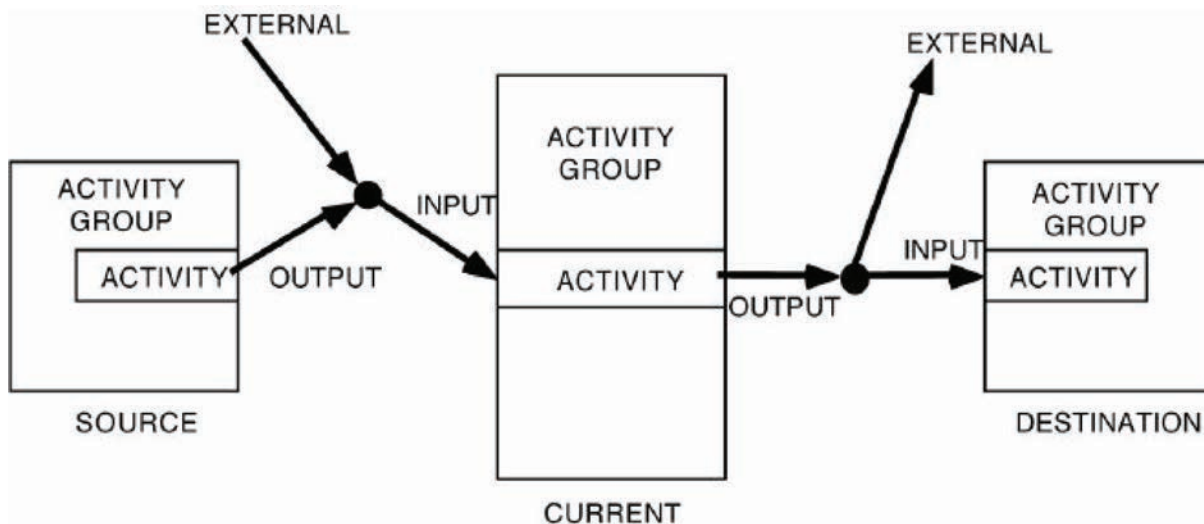


Figure 1 — Activity group

3.74

activity identifier

1. a short, unique numeric or text identification assigned to each schedule activity to differentiate that project activity from other activities. Typically unique within any one project schedule network diagram [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.75

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® Guide) — Fifth Edition]

3.76

activity resource requirements

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® Guide) — Fifth Edition]

3.77

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]

3.78

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

3.79

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® Guide Fifth Edition]

3.80**actor**

1. role (with respect to that action) in which the enterprise object fulfilling the role participates in the action [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language*, 6.3.2] **2.** organization or CASE tool that supplies or acquires SEE services [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services*, 2.10] **3.** in UML, someone or something outside the system that interacts with the system

Note 1 to entry: It can be of interest to specify which actor initiates that action.

3.81**actual cost (AC)****actual cost of work performed (ACWP)**

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® Guide) — Fifth Edition*]
cf. earned value management, earned value technique

EXAMPLE 1

3.82**actual depreciation**

1. true loss in value of an asset, determined only when the asset is sold

3.83**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
cf. constant dollar analysis

3.84**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® Guide) — Fifth Edition*]

3.85**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]

EXAMPLE outputs to hardware, changes to data, reports, and communication messages sent

3.86**ACWP**

1. actual cost of work performed

3.87**adaptability**

1. degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.8.1]
cf. flexibility

Note 1 to entry: Adaptability includes the scalability of internal capacity, such as screen fields, tables, transaction volumes, and report formats. Adaptations include those carried out by specialized support staff, business or operational staff, or end users. If the system is to be adapted by the end user, adaptability corresponds to suitability for individualization as defined in ISO 9241-110.

3.88**adaptation data**

1. data used to adapt a program to a given installation site or to given conditions in its operational environment

3.89**adaptation parameter**

1. variable that is given a specific value to adapt a program to a given installation site or to given conditions in its operational environment

EXAMPLE the variable `Installation_Site_Latitude`

3.90**adapter**

1. object adapter [ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 3.2.1]

3.91**adaptive life cycle**

1. a project life cycle, also known as change-driven or agile methods, that is intended to facilitate change and require a high degree of ongoing stakeholder involvement. Adaptive life cycles are also iterative and incremental, but differ in that iterations are very rapid (usually 2-4 weeks in length) and are fixed in time and resources [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.92**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]

EXAMPLE The operating system is upgraded and some changes are made to accommodate the new operating system.

Note 1 to entry: 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.

3.93**added source statements**

1. count of source statements that were created specifically for the software product

3.94**additional quality planning tools**

1. a set of tools used to define the quality requirements and to plan effective quality management activities. They include, but are not limited to: brainstorming, force field analysis, nominal group techniques and quality management and control tools [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.95**additive weighting**

1. assignment of different values to increase the importance of selected decision attributes
cf. compensatory decision technique, nondimensional scaling, analytic hierarchy process

3.96**address**

1. number, character, or group of characters that identifies a given device or storage location **2.** to refer to a device or storage location by an identifying number, character, or group of characters **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.

3.97**address field****address part**

1. field of a computer instruction that contains addresses, information necessary to derive addresses, or values of operands
cf. operation field

3.98**address format**

1. number and arrangement of address fields in a computer instruction **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
cf. n-address instruction, n-plus-one address instruction

3.99**address modification**

1. arithmetic, logical, or syntactic operation performed on an address
cf. effective address, indexed address, relative address, relocatable address

3.100**address space**

1. addresses that a computer program can access **2.** number of memory locations that a central processing unit can address

Note 1 to entry: 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.

3.101**addressing exception**

1. exception that occurs when a program calculates an address outside the bounds of the storage available to it
cf. data exception, operation exception, overflow exception, protection exception, underflow exception

3.102**addressing mode**

1. method to search operand position in the instruction set architecture for a central processing unit

3.103**addressing range**

1. address space specified and used by the instruction system of a computer

Note 1 to entry: An addressing range depends on the bits of address lines and addressing mode.

3.104**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 1 to entry: This measure does not represent functional size.

3.105**adjusting leads and lags**

1. a technique used to find ways to bring project activities that are behind into alignment with plan during project execution [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.106**ADM**

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

3.107**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]

3.108**ADT**

1. Abstract Data Type [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.3]

3.109**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]

3.110**adverse consequence**

1. undesirable consequence associated with a loss [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.4.2] **2.** consequence that results in a specified level of loss [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.1]
cf. risk

Note 1 to entry: An adverse consequence results from the system-of-interest being in a dangerous condition combined with the environment of the system being in its worst-case state (relative to the adverse consequence). The concept of adverse consequences covers not only harm in the safety context, but also other losses, such as loss of assets in the security context.

3.111**AE(I)**

1. Application Entity (Invocation) [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.112**affective relationship**

1. functional dependency between prior input interaction occurrences and later output interaction occurrences in a behavior pattern [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections — Reference Model for Specifying System Behavior*, 3.2]

3.113**afferent**

1. pertaining to a flow of data or control from a subordinate module to a superordinate module in a software system
cf. efferent

3.114**affinity diagram**

1. a group creativity technique that allows large numbers of ideas to be classified into groups for review and analysis [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.115**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 1 to entry: 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.

3.116**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]

3.117**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]

3.118**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 1 to entry: Aggregation methods are based on compensatory or non-compensatory models.

3.119**agile development**

1. software development approach based on iterative development, frequent inspection and adaptation, and incremental deliveries, in which requirements and solutions evolve through collaboration in cross-functional teams and through continuous stakeholder feedback [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.4; ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.1]

3.120**agile environment**

1. organization or team implementing agile development methods and approaches [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.2]

3.121**agreement**

1. mutual acknowledgment of terms and conditions under which a working relationship is conducted [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.5; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.4] **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® Guide) — Fifth Edition]cf. contract

3.122**AHP**

1. analytic hierarchy process

3.123**algebraic language**

1. programming language that permits the construction of statements resembling algebraic expressions, such as $Y = X + 5$ *cf.* algorithmic language, list processing language, logic programming language

EXAMPLE FORTRAN

3.124**algorithm**

1. finite set of well-defined rules for the solution of a problem in a finite number of steps **2.** sequence of operations for performing a specific task **3.** finite ordered set of well-defined rules for the solution of a problem [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE a complete specification of a sequence of arithmetic operations for evaluating sine x to a given precision

3.125**algorithmic language**

1. programming language designed for expressing algorithms

cf. algebraic language, list processing language, logic programming language

EXAMPLE ALGOL

3.126**alias**

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*]

3.127**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*]

cf. developmental configuration, functional baseline, product baseline, allocated configuration identification

3.128**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

cf. functional configuration identification, product configuration identification, allocated baseline

Note 1 to entry: 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.

3.129**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

3.130 allocation

1. process of distributing requirements, resources, or other entities among the components of a system or program **2.** result of the distribution of requirements, resources, or other entities among the components of a system or program

Note 1 to entry: Allocation can be made entirely to hardware, software, or humans, or to some combination to be resolved upon further functional decomposition.

3.131 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 1 to entry: The entitlement schema enables the recording of entitlement allocations.

3.132 alpha testing

1. first stage of testing before a product is considered ready for commercial or operational use
cf. beta testing

Note 1 to entry: often performed only by users within the organization developing the software

3.133 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*]

3.134 ALS

1. Application Layer Structure [ISO/IEC 10746-1:1998 *Information technology — Open Distributed Processing — Reference model: Overview*]

3.135 alternate flow alternate path

1. part of a use case that describes its alternative implementations

Note 1 to entry: It is also used to describe error conditions, since errors can be considered a kind of alternative.

3.136 alternate key

1. candidate key of an entity other than the primary key [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language tax and Semantics for IDEF1X97 (IDEFObject)*, 3.1.5]

Note 1 to entry: [key style]

3.137 alternative analysis

1. a technique used to evaluate identified options in order to select which options or approaches to use to execute and perform the work of the project [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.138 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® Guide) — Fifth Edition]

3.139**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*]

3.140**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*]

EXAMPLE an abacus

3.141**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® Guide) — Fifth Edition]

3.142**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 **2.** investigation and collection phase of development that aims to specify types of users and their information needs [ISO/IEC/IEEE 26512:2011 *Systems and software engineering — Requirements for acquirers and suppliers of user documentation*, 4.2]

EXAMPLE test

3.143**analyst**

1. member of the technical community who is skilled and trained to define problems and to analyze, develop, and express algorithms

EXAMPLE systems engineer, business analyst

3.144**analytic hierarchy process (AHP)**

1. use of matrixes to manage pair-wise relationships in decision-making
cf. additive weighting, nondimensional scaling, compensatory decision technique

3.145**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 [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.146**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 [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]
cf. modifiability

Note 1 to entry: 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.

3.147**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 [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.6]

cf. generic ancestor, reflexive ancestor

3.148**ancestral box**

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

3.149**ancestral diagram**

1. diagram that contains an ancestral box [IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0, 2.1.7]

3.150**anchor point**

1. a milestone in software scheduling at which a major project life cycle transition occurs [Software Extension to the PMBOK® Guide Fifth Edition]

3.151**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

3.152**annotation**

1. further documentation accompanying a requirement **2.** label represented as text near to the object it is associated with [ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.1]

EXAMPLE background information or descriptive material

3.153**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 [ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.3]

EXAMPLE the initiation of the sending of a message in a messaging system

3.154**annual equivalent****AE(i)**

1. representation of a cash flow as a series of equal annual payments (at a stated interest rate) over the planning horizon.

cf. future worth, present worth

3.155**annual percentage rate (APR)**

1. nominal annual interest rate

3.156**annuity**

1. amount of a series of equal payments at regular intervals over a planning horizon

3.157**anomaly**

1. condition that deviates from expectations, based on requirements specifications, design documents, user documents, or standards, or from someone's perceptions or experiences [*IEEE 1028-2008 IEEE Standard for Software Reviews and Audits*, 3.1] **2.** 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 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1]

3.158**ANSI**

1. American National Standards Institute.

3.159**anticipatory buffering**

1. buffering technique in which data are stored in a buffer in anticipation of a need for the data
cf. dynamic buffering, simple buffering

3.160**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
cf. demand paging

3.161**AOA**

1. analysis of alternatives [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.162**AON**

1. Activity-on-Node

3.163**AP(I)**

1. Application Process (Invocation) [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.164**aperiodic task****asynchronous task**

1. task activated on demand

3.165**API**

1. Application Program Interface [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.166**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]

3.167**application
application system**

1. system for collecting, saving, processing, and presenting data by means of a computer [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.1; ISO/IEC 24570:2005 *Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*] **2.** a coherent collection of automated procedures and data supporting a business objective [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10] **3.** 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] cf. information system

EXAMPLE accounts payable, accounts receivable, payroll, procurement, shop production, assembly line control, air search radar, target tracking, weapons firing, flight line scheduling and passenger reservations

Note 1 to entry: 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.

3.168**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]

3.169**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 1 to entry: The application architecture captures the high-level design of a specific member product of a product line.

3.170**application area**

1. a category of projects that have common components significant in such projects, but are not needed or present in all projects. Application areas are usually defined in terms of either the product (i.e., by similar technologies or production methods) or the type of customer (i.e., internal versus external, government versus commercial) or industry sector (i.e., utilities, automotive, aerospace, information technologies, etc.) Application areas can overlap. [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.171**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 1 to entry: Application asset encompasses requirements, an architectural design, components, and tests.

3.172**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]

3.173**application boundary**

1. the border between the application and its environment of other applications and users [ISO/IEC 24570:2005 *Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.174**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]

3.175**application engineering**

1. process of constructing or refining application systems by reusing assets [IEEE 1517-2010 *IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes*, 3] **2.** 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]

3.176**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]

3.177**application frameworks**

1. subsystem design made up of a collection of abstract and concrete classes and interfaces between them

Note 1 to entry: Frameworks are often instantiation of a number of patterns.

3.178**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.** the size of an application expressed in function points [ISO/IEC 24570:2005 *Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*] **3.** 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 1 to entry: 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.

3.179**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]

3.180**application generator**

1. code generator that produces programs to solve one or more problems in a particular application area

EXAMPLE a payroll generator

3.181**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 1 to entry: 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.

3.182**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 1 to entry: An application management organization can be an internal or external unit in relation to the user organization.

3.183**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]

EXAMPLE Programs, sources, databases, documentation, data structures, test files, and scripts.

3.184**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 1 to entry: 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.

3.185**application problem**

1. problem submitted by an end user and requiring information processing for its solution. ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.186**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]

3.187**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]

3.188**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]

3.189**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]

3.190**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]

3.191**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]

3.192**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 **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 **4.** software of an application [ISO/IEC 16350:2015 *Information technology — Systems and software engineering — Application management*, 4.6]

Note 1 to entry: 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.

3.193**application specific integrated circuit (ASIC)**

1. integrated circuit customized for a particular use

3.194**application-oriented language**

1. computer language with facilities or notations applicable primarily to a single application area
cf. authoring language, specification language, query language

EXAMPLE a language for computer-assisted instruction or hardware design

3.195**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]

3.196**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® Guide) — Fifth Edition*]

3.197**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® Guide) — Fifth Edition*]
cf. discrete effort

Note 1 to entry: Apportioned effort is one of three earned value management (EVM) types of activities used to measure work performance.

3.198**appraisal findings**

1. results of an appraisal that identify the most important issues, problems, or opportunities for process improvement within the appraisal scope

Note 1 to entry: Appraisal findings are inferences drawn from corroborated objective evidence.

3.199**appraisal participants**

1. members of the organizational unit who participate in providing information during an appraisal

3.200**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

3.201**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]/cf. functional appropriateness

Note 1 to entry: 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.

3.202**approval**

1. written notification, by an authorized representative, that an information item appears to satisfy requirements and is complete [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.1]

Note 1 to entry: Such approval does not shift responsibility from the supplier to meet requirements under a two-party situation.

3.203**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 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.5.2]

Note 1 to entry: 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.

3.204**approved change request**

1. a change request that has been processed through the integrated change control process and approved [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.205**approved change requests review**

1. a review of the change requests to verify that these were implemented as approved [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.206**approved modification**

1. disposition of one or more proposed changes authorizing change to any SCIs [ISO/IEC TR 15846:1998 *Information technology — Software life cycle processes — Configuration Management*, 4.1]

Note 1 to entry: 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.

3.207**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 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.1]

3.208**arc annotation**

1. expression that can involve constants, variables and operators used to annotate an arc of a net [ISO/IEC 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.1.3]

Note 1 to entry: The expression must evaluate to a multiset over the type of the arc's associated place.

3.209**architect**

1. person, team, or organization responsible for systems architecture

3.210**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 1 to entry: Architecting takes place in the context of an organization or a project.

3.211**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 **2.** the result of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system
cf. functional design

3.212**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

3.213**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

3.214**architectural structure**

1. physical or logical layout of the components of a system design and their internal and external connections

EXAMPLE function-oriented (structured) design, object-oriented design, and data structure-oriented design

3.215

architectural style

1. definition of a family of systems in terms of a pattern of structural organization **2.** characterization of a family of systems that are related by sharing structural and semantic properties

EXAMPLE pipes and filters, layers, rule-based systems, and blackboards

3.216

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 TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.6; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.5; ISO/IEC/IEEE 42010:2011 *Systems and software engineering — Architecture description*, 3.2] **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]
cf. component, module, subprogram, routine

3.217

architecture description (AD)

architectural description

1. work product used to express an architecture [ISO/IEC/IEEE 42010:2011 *Systems and software engineering — Architecture description*, 3.3]

3.218

architecture framework

1. conventions, principles and practices for the description of architectures established within a specific domain of application or community of stakeholders [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.7; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.6; ISO/IEC/IEEE 42010:2011 *Systems and software engineering — Architecture description*, 3.4]

EXAMPLE Generalised Enterprise Reference Architecture and Methodologies (GERAM) [ISO 15704], Reference Model of Open Distributed Processing (RM-ODP) [ISO/IEC 10746]

3.219

architecture view

1. work product expressing the architecture of a system from the perspective of specific system concerns [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.8; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.7; ISO/IEC/IEEE 42010:2011 *Systems and software engineering — Architecture description*, 3.5]

cf. architecture viewpoint

3.220

architecture viewpoint

1. work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.9; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.8; ISO/IEC/IEEE 42010:2011 *Systems and software engineering — Architecture description*, 3.6]

cf. architecture view

3.221**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 1 to entry: ADM does not preclude source-to-source migrations (where appropriate), but encourages user organizations to consider modernization from an analysis and design perspective.

3.222**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 1 to entry: Due to technology upgrades, some archival pages cannot be readily rendered unless they are upgraded along with active pages

3.223**argument**

1. independent variable **2.** specific value of an independent variable **3.** constant, variable, or expression used in a call to a software module to specify data or program elements to be passed to that module

EXAMPLE the variable m in the equation $E = mc^2$

3.224**argument sort****input sort**

1. sort of an argument of an operator [ISO/IEC 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.23.1]

3.225**arity**

1. number of roles that participate in a relationship [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2] **2.** input sorts and output sort for an operator [ISO/IEC 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.2]

Note 1 to entry: A binary relationship has an arity of two. An n -ary relationship has an arity of n . ($n > 2$) sometimes known as the "degree" of a relationship.

3.226**arranging**

1. activity of sequencing attributes in a transactional function [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.5]

3.227**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

EXAMPLE a matrix, table, or vector

3.228**arrow**

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

cf. arrow segment, boundary arrow, internal arrow

3.229**arrow label**

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 1 to entry: Specifically, an arrow label identifies the object type set that is represented by an arrow segment.

3.230**arrow meaning**

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]

EXAMPLE a physical thing, a data element

3.231**arrow role**

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 1 to entry: There are four arrow roles: input, control, output, and mechanism.

3.232**arrow segment**

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]

3.233**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 1 to entry: An enterprise object that is an artifact in one action can be an actor in another action.

3.234**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]

3.235**ASIC**

1. application specific integrated circuit

3.236**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]

3.237**ASO**

1. Application Service Object [ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview]

3.238**aspect**

1. special consideration within product line engineering process groups and tasks to associated specialized methods and tools [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.7*]

3.239**ASR**

1. alternative systems review [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.240**assemble**

1. to translate a computer program expressed in an assembly language into its machine language equivalent 2. process of constructing from parts one or more identified pieces of software [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]

cf. compile, disassemble, interpret

3.241**assemble-and-go**

1. operating technique in which there are no stops between the assembling, linking, loading, and execution of a computer program

3.242**assembled origin**

1. address of the initial storage location assigned to a computer program by an assembler, a compiler, or a linkage editor

cf. loaded origin, offset (1), starting address

3.243**assembler**

1. computer program that translates programs expressed in assembly language into their machine language equivalents

cf. absolute assembler, compiler, cross-assembler, interpreter, relocating assembler

3.244**assembly**

1. collection of units in which ports on different units are compatibly interconnected so behaviors of one unit can influence behaviors of another unit by means of interactions occurring through interconnected ports [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections — Reference Model for Specifying System Behavior, 3.3*]

Note 1 to entry: An assembly can be regarded as behaviorally equivalent with a unit. The ports of the unit are just those ports of the assembly units that are not internally interconnected. The behavior of the unit is a composition of the shared behaviors of the assembly units

3.245**assembly code****assembler code**

1. computer instructions and data definitions expressed in a form that can be recognized and processed by an assembler

cf. compiler code, interpretive code, machine code

3.246

assembly language
assembler language
low-level language
second-generation 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

cf. fifth-generation language, fourth-generation language, high order language, machine language.

3.247

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. **2.** function or macro that complains loudly if a design assumption on which the code is based is not true

cf. invariant, proof of correctness

Note 1 to entry: Types include input assertion, loop assertion, output assertion.

3.248

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]

3.249

assessment body

1. body that performs an assessment [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.2.1]

Note 1 to entry: A body can be an organization or part of an organization that performs the assessment.

3.250

assessment constraint

1. restriction placed on the use of the assessment outputs or on the assessment team's freedom of choice regarding the conduct of the assessment [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.2.2]

3.251

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]

EXAMPLE work products, practice, or resource

3.252

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]

3.253

assessment output

1. tangible results from an assessment [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.2.4]

cf. assessment record

3.254

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]



Documents

Document Sharing Hub

EXAMPLE the assessment sponsor, assessors, and organizational unit members

3.255

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]

3.256

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]

3.257

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]

3.258

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]

3.259

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]

3.260

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]

3.261

asset

1. 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] 2. item that has been designed for use in multiple contexts [IEEE 1012-2012 *IEEE Standard for System and Software 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 1012-2012 *IEEE Standard for System and Software Verification and Validation*, 3.1] 4. item, thing, or entity that has potential or actual value to an organization [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.2]

EXAMPLE requirements documents, source code modules, measurement definitions

Note 1 to entry: For most organizations, 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.

3.262

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*, 8.6]

3.263**asset management**

1. coordinated activities of an organization to realize value from assets [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.3]

3.264**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]

3.265**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 1 to entry: 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.

3.266**assignment**

1. for a set of variables, the association of a value (of correct type) to each variable [ISO/IEC 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.3]

cf. binding

3.267**assignment statement**

1. computer program statement that assigns a value to a variable

cf. control statement, declaration, clear, initialize, reset

EXAMPLE $Y = X - 5$

3.268**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]

3.269**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]

EXAMPLE Braille displays, screen readers, screen magnification software, and eye tracking devices

3.270**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
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]

3.271**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]

3.272**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]

3.273**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 1 to entry: An associative entity is used when a relationship does not otherwise provide sufficient mechanisms.

3.274**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]

cf. entity type

3.275**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 1 to entry: The form of expression used to state an associative literal is className with propertyName: propertyValue.

3.276**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® Guide) — Fifth Edition]

3.277**assumptions analysis**

1. a technique that explores the accuracy of assumptions and identifies risks to the project from inaccuracy, inconsistency, or incompleteness of assumptions [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.278**assurance**

1. grounds for justified confidence that a claim has been or will be achieved [ISO/IEC 15026-1:2013 *Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.1.1]

3.279**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 15026-1:2013 *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 1 to entry: 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

3.280**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]
cf. ensure

3.281**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*]

3.282**asynchronous communication interface adapter**

1. functional unit to connect interfaces for asynchronous communications

3.283**asynchronous communication interface adapter (ACIA)**

1. functional unit to connect interfaces for asynchronous communications

3.284**asynchronous I/O device**

1. I/O device that generates an interrupt after producing some input or generating some output

3.285**asynchronous I/O device interface task**

1. task that interfaces to an I/O device and is activated by interrupts from that device

3.286**asynchronous message communication****loosely coupled message communication**

1. communication in which a producer task sends a message to a consumer task and does not wait for a response

Note 1 to entry: A message queue could build up between the tasks.

3.287**atomic type****primitive type**

1. data type, each of whose members consists of a single, nondecomposable data item
cf. composite type

3.288**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]

3.289**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]

3.290**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 15026-1:2013 *Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.4.6]

3.291**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/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.2; ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.1] **3.** measurable physical or abstract property of an entity [IEEE 1061-1998 (R2004) *IEEE Standard for Software Quality Metrics Methodology*, 2.1] **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.** unique item of information about an entity [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10] **7.** single-valued characteristic of an entity or relationship [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2] **8.** 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]

EXAMPLE people, objects, places, events, ideas, or combinations of things

Note 1 to entry: 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).

3.292**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]

3.293**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. After each unit is inspected, the decision is made to accept a lot, reject it, or inspect another unit [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.294**attributed relationship**

1. relationship that has attributes [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2]

3.295**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]

cf. entity

3.296**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 1 to entry: There can be different audiences for documentation (for example, management, data entry, maintenance).

3.297**audience research**

1. planned process of interviews of representative users and analysis of interview records and personnel records

Note 1 to entry: The purpose of audience research is to determine the abilities, training, experience, limitations, prejudices and preferences of the intended readers of a document.

3.298**audit**

1. 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] **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 TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.10; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.9] **3.** independent examination of a software product, software process, or set of software processes to assess compliance with specifications, standards, contractual agreements, or other criteria [IEEE 1028-2008 *IEEE Standard for Software Reviews and Audits*, 3.2] **4.** independent assessment of products and processes, conducted by an authorized person to assess compliance with requirements [ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes*, 4.5] **5.** 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 29110-2-1:2015 — *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.7]

Note 1 to entry: An audit results in a clear indication of whether the audit criteria have been met.

3.299**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 1 to entry: One auditor of the audit team is appointed as the audit team leader. The audit team can include auditors-in-training.

3.300**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]

3.301**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]

3.302**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]

3.303**authoring language**

1. high-level programming language used to develop courseware for computer-assisted instruction
cf. authoring system

3.304**authoring system**

1. programming system that incorporates an authoring language

3.305**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® Guide) — Fifth Edition*]

3.306**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 1 to entry: Unlike a permission, an authorization is an empowerment.

3.307**automate**

1. to make a process or equipment automatic [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.308**automated systems process****assisted process****assisted software process****assisted systems process****automated process****automated software 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*, 2.9]

3.309**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 **2.** software tool that automates part or all of the verification process

3.310**automatic**

1. pertaining to a process or equipment that, under specified conditions, functions without human intervention [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.311**automation**

1. conversion of processes or equipment to automatic operation, or the results of the conversion [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.312**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 [ISO/IEC TR 29110-3-4:2015 *Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-4: Autonomy-based improvement method*, 3.2] **2.** motivated professional process improvement with understanding work (process) objectives, technology status quo, and outcomes from product use, not forced by anybody [ISO/IEC 29110-2-1:2015 — *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.11]

3.313**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 **2.** degree to which a system or component is operational and accessible when required for use [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.5.2] **3.** ability of an application object to perform its required function at an agreed instant or over an agreed period of time [ISO/IEC 16350-2015 *Information technology — Systems and software engineering — Application management*, 4.7]

cf. error tolerance, fault tolerance, reliability, robustness

Note 1 to entry: Availability is normally expressed as a ratio or percentage of the time that the service or service component is actually available for use by the customer to the agreed time that the service should be available. Availability is a combination of maturity (which reflects the frequency of 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.

3.314**available interaction**

1. interaction that is allowed by one unit and controlled by another unit [IEEE 1175.4-2008 *IEEE Standard for CASE Tool Interconnections — Reference Model for Specifying System Behavior*, 3.4]

3.315**BAC**

1. budget at completion [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.316**back matter**

1. material that appears at the end of printed documentation, such as an index

3.317**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]

cf. mutation testing

3.318**background**

1. in job scheduling, the computing environment in which low-priority processes or those not requiring user interaction are executed

cf. foreground. background processing

3.319**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

cf. foreground processing

3.320**backlog**

1. a listing of product requirements and deliverables to be completed, written as stories, and prioritized by the business to manage and organize the project's work [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** a set of software features awaiting development in a subsequent iteration [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.321**backout**

1. to undo the effects of a commit

Note 1 to entry: often by introducing a new commit that restores things to their previous state

3.322**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 **2.** to create or designate a system, component, file, procedure, or person as a replacement

3.323**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]

3.324**backup programmer**

1. assistant leader of a chief programmer team

cf. chief programmer

Note 1 to entry: 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.

3.325**Backus-Naur Form**

1. formal meta-language used for defining the syntax of a language in a textual format [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.1]

3.326**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® Guide) — Fifth Edition*]

cf. schedule network analysis