

INTERNATIONAL
STANDARD

ISO/IEC/
IEEE
24765

Second edition
2017-09

Systems and software engineering — Vocabulary

Ingénierie des systèmes et du logiciel — Vocabulaire



Reference number
ISO/IEC/IEEE 24765:2017(E)

© ISO/IEC 2017
© IEEE 2017



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2017, Published in Switzerland

© IEEE 2017

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from ISO, IEC or IEEE at the respective address below.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Institute of Electrical and Electronics Engineers, Inc
3 Park Avenue, New York
NY 10016-5997, USA

stds.ipr@ieee.org
www.ieee.org

© ISO/IEC 2017 – All rights reserved
© IEEE 2017 – All rights reserved

Contents

Page

Foreword	vi
Introduction.....	viii
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.....	516
Bibliography	517

List of Figures

Figure 1 — Activity group.....	10
Figure 2 — Bathtub curve.....	43
Figure 3 — Block diagram	47
Figure 4 —Box Diagram	50
Figure 5 —Bubble chart	52
Figure 6 —Call graph.....	57
Figure 7 —Case construct.....	59
Figure 8 — Category.....	60
Figure 9 — Data flow diagram	116
Figure 10 — Data structure diagram.....	119
Figure 11 — Directed graph	140
Figure 12 — Documentation tree	144
Figure 13 — Flowchart.....	186
Figure 14 — If-then-else construct.....	213
Figure 15 — Input-process-output chart.....	227
Figure 16 — Modification request.....	281
Figure 17 — Structure chart.....	443
Figure 18 — UNTIL construct.....	491
Figure 19 — Waterfall model	509
Figure 20 — Website.....	510
Figure 21 — WHILE construct.....	511

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

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.

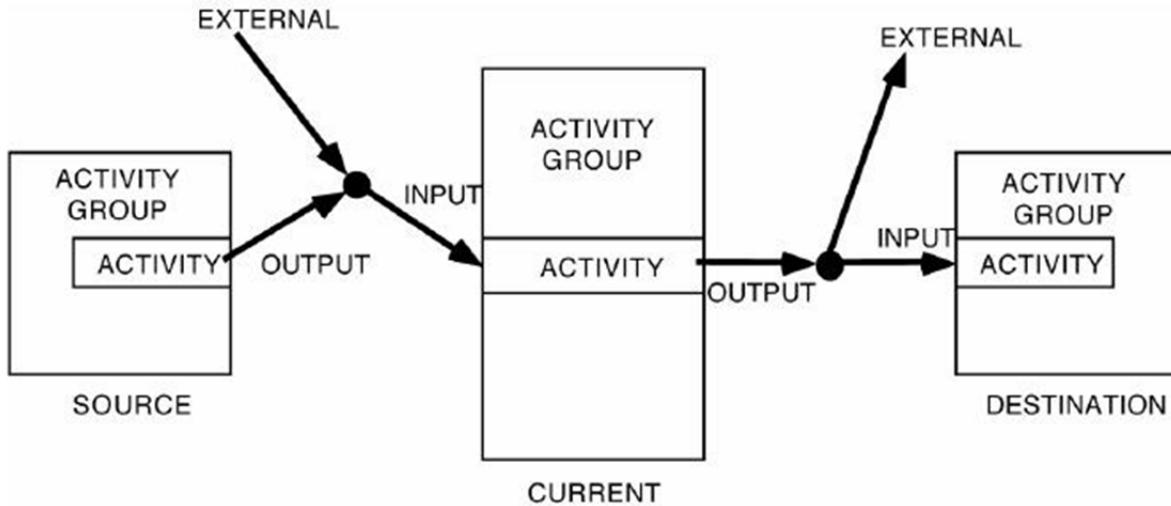


Figure 1 — Activity group

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]

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*, 3.6]

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

34

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

3.327

backward recovery

- 1.** reconstruction of a file to a given state by reversing all changes made to the file since it was in that state **2.** type of recovery in which a system, program, database, or other system resource is restored to a previous state in which it can perform required functions

cf. forward recovery

3.328

bag

- 1.** collection class whose members are unordered but in which duplicates are meaningful [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.11]

cf. list, set

3.329

ball grid array (BGA)

- 1.** surface-mounted integrated circuit package with multiple connections in a grid pattern on the bottom surface

Note 1 to entry: provides more connections than on packages with connectors on the edges only

3.330

bar chart

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

cf. Gantt chart

3.331

base address

- 1.** address used as a reference point to which a relative address is added to determine the address of the storage location to be accessed

cf. indexed address, relative address, self-relative address

3.332

base class

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

cf. derived class

3.333

base functional component (BFC)

functional service

- 1.** an elementary unit of functional user requirements defined by and used by an FSM method for measurement purposes [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.1; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.1; ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.8*]

EXAMPLE: A functional user requirement "Maintain Customers" consists of the following BFCs: "Add a new customer", "Report Customer Purchases", and "Change Customer Details". A collection of logically related business data is maintained by the software as "Customer Details".

3.334

base functional component type (BFC type)

- 1.** defined category of Base Functional Component [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.2*]

3.335

base measure

- 1.** measure defined in terms of an attribute and the method for quantifying it [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.3*]

Note 1 to entry: A base measure is functionally independent of other measures.

3.336

base practice (BP)

- 1.** activity that, when consistently performed, contributes to achieving a specific process purpose [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.3.2*]

3.337

base standard

- 1.** approved International Standard or Telecommunication Standardization Sector of the International Telecommunications Union (ITU-T) Recommendation [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 2.5*]

3.338

base value

base choice

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

3.339

baseline

1. formally approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle [ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.10] 2. specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures [ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes*, 4.6] 3. agreement or result designated and fixed at a given time, from which changes require justification and approval [IEEE 1012-2012 *IEEE Standard for System and Software Verification and Validation*, 3.1] 4. snapshot of the state of a service or individual configuration items at a point in time [ISO/IEC 19770-1:2012 *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3.1] 5. formally controlled and maintained set of data that serves as the basis for defining change [IEEE 15288.1:2014 *IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.1] 6 [verb] to establish and approve a set of data [IEEE 15288.1:2014 *IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.1] 7. the approved version of a work product that can be changed only through formal change control procedures and is used as a basis for comparison [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 8. approved version of a configuration item, regardless of media, formally designated and fixed at a specific time during the configuration item's life cycle [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.11]

Note 1 to entry: Some baselines are project deliverables, while others provide the basis for further work. A baseline, together with all approved changes to the baseline, represents the current approved configuration. The term is thus used to refer to a particular version of a software configuration item that has been agreed on, e.g., as a stable base for further development or to mark a specific project milestone. In either case, any new baseline is agreed through the project's agreed change control procedures.

3.340

baseline design

1. system design that has been agreed on by all stakeholders interested in the system development

3.341

baseline document

1. system or software document that defines a work product that has been placed under configuration management

EXAMPLE: design specifications, requirements specifications, system specifications

3.342

baseline function point count

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

3.343

baseline management

1. in configuration management, the application of technical and administrative direction to designate the documents and changes to those documents that formally identify and establish baselines at specific times during the life cycle of a configuration item

3.344

basic engineering object

BEO

1. engineering object that requires the support of a distributed infrastructure [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.1]

3.345

basic flow

basic path

happy day scenario

1. part of a use case that describes its most common implementation

Note 1 to entry: The basic flow is written assuming that no errors or alternatives exist.

3.346

basic interworking facility

1. set of service primitives which have a direct correspondence with computational signals which model computational operations [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions*, 3.3.4]

3.347

basic maturity level

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

3.348

basic process set

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

3.349

basic profile

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

3.350

basic symbol

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

3.351

basis of estimates

1. supporting documentation outlining the details used in establishing project estimates such as assumptions, constraints, level of detail, ranges, and confidence levels [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.352

basis set

1. set of objects used to create a multiset [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.4]

3.353

batch

1. pertaining to a system or mode of operation in which inputs are collected and processed all at one time, rather than being processed as they arrive, and a job, once started, proceeds to completion without additional input or user interaction

cf. conversational, interactive, online, real time

3.354

bathtub curve

1. graph of the number of failures in a system or component as a function of time

Note 1 to entry: The name is derived from the usual shape of the graph: a period of decreasing failures (the early-failure period), followed by a relatively steady period (the constant-failure period), followed by a period of increasing failures (the wearout-failure period).

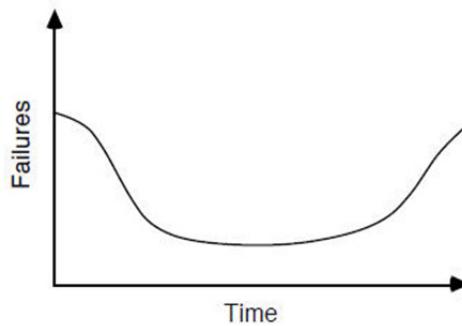


Figure 2 — Bathtub curve

3.355

Bayes' rule

1. statistical formula that relates the conditional probability $P(A | B)$ to the inverse conditional probability $P(B | A)$

3.356

BCWP

1. budgeted cost of work performed

BCWS

1. budgeted cost of work scheduled

3.357

behavior

1. observable activity of a system, measurable in terms of quantifiable effects on the environment whether arising from internal or external stimulus [IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.1] 2. peculiar reaction of a thing under given circumstances [IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.1] 3. aspect of an instance's specification that is determined by the state-changing operations it can perform [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.12] 4. of an object, a collection of actions with a set of constraints on when they may occur [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.7] 5. aggregate of all unit responses to internal and external stimuli [IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.5]

3.358

behavior pattern

1. relationship that maps a sequence of stimulus interactions to a sequence of response interactions [IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.6]

3.359

behavior specification

1. structured collection of data that describes the potential variety of behavior possible from a system [IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.2]

3.360

behavior state

1. state which represents the partitioning of all unit behavior patterns into (possibly overlapping) subsets of behavior patterns that can be elicited from a unit at a particular point in time [IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.7]

3.361

behavioral compatibility

1. identical behavior of two objects, such that one object can replace the other with respect to a set of criteria without the environment being able to notice the difference in the objects' behavior on the basis of the set of

criteria [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.4]

3.362

benchmark

- 1.** standard against which results can be measured or assessed [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.2] **2.** procedure, problem, or test that can be used to compare systems or components to each other or to a standard **3.** reference point against which comparisons can be made [ISO/IEC 29155-1:2011 *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.1]

3.363

benchmarking

- 1.** activity of comparing objects of interest to each other or against a benchmark to evaluate characteristic(s) [ISO/IEC 29155-1:2011 *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.2] **2.** the comparison of actual or planned practices, such as processes and operations, to those of comparable organizations to identify best practices, generate ideas for improvement, and provide a basis for measuring performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: In the context of ISO/IEC 29155, the object of interest is IT project performance, and the characteristic is a particular aspect of an IT project such as productivity.

3.364

benchmarking analyst

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

3.365

benchmarking experience base

- 1.** information store that contains the evaluation of the information products and the benchmarking activity, as well as any lessons learned during benchmarking and analysis [ISO/IEC 29155-1:2011 *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.3]

3.366

benchmarking method

- 1.** logical sequence of general steps to describe the process of comparing one or more attributes against a reference attribute with respect to a specified scale [ISO/IEC 29155-1:2011 *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.4]

3.367

benchmarking report

- 1.** document of the results of an instance of benchmarking [ISO/IEC 29155-3:2015 *Systems and software engineering — Information technology project performance benchmarking framework — Part 3: Guidance for reporting*]

Note 1 to entry: Document usually consists of various formats (e.g. textual descriptions, numeric values, statistical charts and tables), and is exchanged via various media (e.g. electronic documents, electronic data set, printed documents, and embedded data within specific computer software).

3.368

benchmarking user

- 1.** person or organization that utilizes the outcome of benchmarking [ISO/IEC 29155-1:2011 *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.5]

3.369

benefit cost analysis

- 1.** in not-for-profit decision analysis, evaluating the desirability of an alternative on the ratio of the net benefits to the population to the net costs to the sponsor

3.370

BEO

- 1.** Basic Engineering Object [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview; ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications, 4*]

3.371

beta test

- 1.** second stage of testing when a product is in limited production use
cf. alpha testing

Note 1 to entry: often performed at a customer site

3.372

BFC

- 1.** base functional component [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 4; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.1*]

BFC class

- 1.** defined group of BFC types [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, 3.1*]

3.373

BFC Type

- 1.** a defined category of BFCs [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.2*]

EXAMPLE: 'External Inputs', 'External Outputs' and 'Logical Transactions', and data stores such as 'Internal Logical Files'

3.374

BGA

- 1.** ball grid array

3.375

bidder conference

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

3.376

bidirectional traceability

- 1.** association among two or more logical entities that is discernible in either direction (to and from an entity)
cf. requirements traceability

3.377

big-bang testing

- 1.** type of integration testing in which software elements, hardware elements, or both are combined all at once into an overall system, rather than in stages

3.378

bill of materials (BOM)

- 1.** documented formal hierarchical tabulation of the physical assemblies, subassemblies, and components needed to fabricate a product

3.379

binary digit (bit)

1. unit of information that can be represented by either a zero or a one. 2 element of computer storage that can hold a unit of information as in (1) 3. numeral used to represent one of the two digits in the binary numeration system; zero (0) or one (1)

3.380

bind

1. to assign a value to an identifier
cf. dynamic binding, static binding

EXAMPLE: to assign a value to a parameter or to assign an absolute address to a symbolic address in a computer program

3.381

binder

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

3.382

binding

1. contractual context resulting from a given establishing behavior [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.5.2] 2. task to make a decision on relevant variants, which will be application assets, from domain assets using the domain variability model and from application assets using the application variability model [ISO/IEC 26550:2015 *Software and systems engineering — Reference model for product line engineering and management*, 3.8]
cf. assignment

3.383

binding behavior

1. establishing behavior between two or more interfaces, and hence between their supporting objects [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.5.1]

3.384

binding endpoint identifier

1. identifier, in the naming context of a capsule, used by a basic engineering object to select one of the bindings in which it is involved, for the purpose of interaction [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.15]

EXAMPLE: a memory address (for a data structure representing an engineering interface)

Note 1 to entry: The same form of binding endpoint identifier can be used, whether the binding involved is either local or distributed.

3.385

binding object

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

3.386

binding precondition

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

3.387

binding time

1. moment of variability resolution [ISO/IEC 26555:2015 *Software and systems engineering — Tools and methods for product line technical management*, 3.3]

3.388

bit

1. either of the digits 0 or 1 when used in the binary numeration system [ISO/IEC 2382:2015, *Information technology — Vocabulary*] 2. binary digit 3. built-in test [IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2]

3.389

bit steering

immediate control

1. microprogramming technique in which the meaning of a field in a microinstruction is dependent on the value of another field in the microinstruction

cf. residual control, two-level encoding

3.390

black box

1. system or component whose inputs, outputs, and general function are known but whose contents or implementation are unknown or irrelevant 2. pertaining to an approach that treats a system or component whose inputs, outputs, and general function are known but whose contents or implementation are unknown or irrelevant

cf. glass box

3.391

block

1. group of contiguous storage locations, computer program statements, records, words, characters, or bits that are treated as a unit 2. to form a group

cf. block-structured language, delimiter

3.392

block diagram

configuration diagram

system resources chart

1. diagram of a system in which the principal parts or functions are represented by blocks connected by lines that show the relationships of the blocks [ISO/IEC 2382:2015, *Information technology — Vocabulary*] 2. diagram of a system, computer, or device in which the principal parts are represented by suitably annotated geometrical figures to show both the functions of the parts and their functional relationships

cf. box diagram, bubble chart, flowchart, graph, input-process-output chart, structure chart

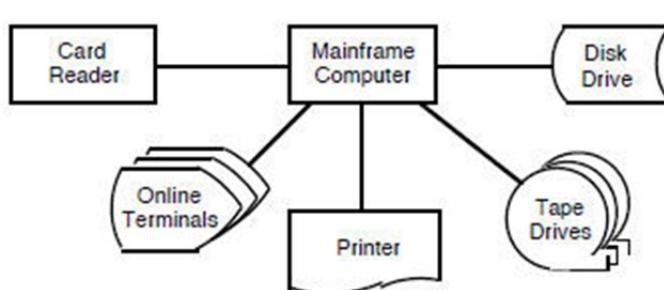


Figure 3 — Block diagram

3.393

block-structured language

1. design or programming language in which sequences of statements, called blocks, are defined, usually with begin and end delimiters, and variables or labels defined in one block are not recognized outside that block
cf. structured programming language

EXAMPLE: Ada, ALGOL, PL/I

3.394

blocking factor

- 1.** number of records, words, characters, or bits in a block

3.395

BMT

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

3.396

body metadata

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

3.397

body of knowledge

- 1.** collection of knowledge items or areas generally agreed to be essential to understanding a particular subject [*ISO/IEC 24773:2008 Software engineering -Certification of software engineering professionals -Comparison framework, 3.1*]

3.398

BOM

- 1.** bill of materials

3.399

Boolean expression

- 1.** expression that evaluates to true or false [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.26.1*]

3.400

Boolean signature

- 1.** signature where one of the sorts is Bool, corresponding to the carrier Boolean in any associated algebra, and one of the constants is true sub Bool, corresponding to the value true in the algebra [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.22.1*]

3.401

boot

- 1.** to initialize a computer system by clearing memory and reloading the operating system

Note 1 to entry: derived from bootstrap

3.402

boot mode

- 1.** initialized mode of program operations when a computer is turned on

3.403

bootstrap

- 1.** short computer program that is permanently resident or easily loaded into a computer and whose execution brings a larger program, such as an operating system or its loader, into memory **2.** to use a program to bring up a larger program, such as an operating system

3.404

bootstrap loader

- 1.** short computer program used to load a bootstrap

3.405

bottom-up

1. pertaining to an activity that starts with the lowest-level components of a hierarchy and proceeds through progressively higher-levels **2.** pertaining to a method or procedure that starts at the lowest level of abstraction and proceeds towards the highest level [*ISO/IEC 2382:2015, Information technology — Vocabulary*] cf. top-down, critical piece first

EXAMPLE: bottom-up design, bottom-up testing

3.406

bottom-up design

1. design approach in which low-level pieces of a system are combined into an overall design. **2.** process of designing a system by identifying low-level components, designing each component separately, and then designing a structure to integrate the low-level components into larger and larger subsystems until the design is finished

3.407

bottom-up estimating

1. a method of estimating project duration or cost by aggregating the estimates of the lower-level components of the work breakdown structure (WBS) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.408

boundary

1. conceptual interface between the software under study and its users [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.9; ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, 3.2] **2.** conceptual interface between the software being measured and its functional users [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.3] **3.** collection of all interface ports between a unit and its environment, characterized by the input and output interactions that it allows [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.8*]**

Note 1 to entry: The boundary provides the measurement analyst(s) with a solid delimiter to distinguish, without ambiguity, what is included inside the measured software from what is part of the measured software's operating environment.

3.409

boundary arrow

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

3.410

boundary ICOM code

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

3.411

boundary value

1. data value that corresponds to a minimum or maximum input, internal, or output value specified for a system or component
cf. stress testing

3.412

box

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

3.413

box detail reference

- square enclosure encompassing a box number, which indicates that the box is decomposed or detailed by a child diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.17*]

3.414

box diagram

Chapin chart

Nassi-Shneiderman chart

- control flow diagram consisting of a rectangle that is subdivided to show sequential steps, if-then-else conditions, repetition, and case conditions

cf. block diagram, bubble chart, flowchart, graph, input-process-output chart, program structure diagram, structure chart

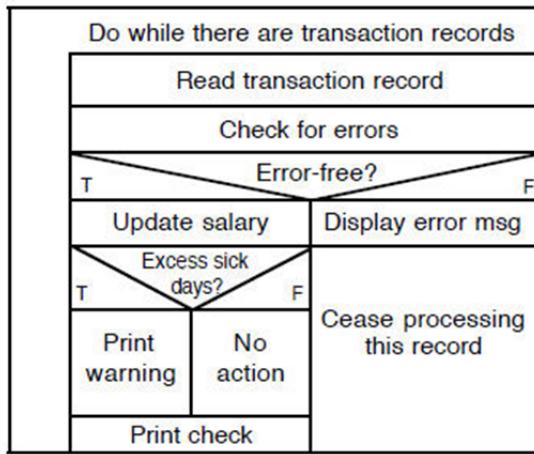


Figure 4 —Box Diagram

3.415

box ICOM code

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

3.416

box name

- verb or verb phrase placed inside a box that names the modeled function [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.19*]
- cf.* function name

Note 1 to entry: A box takes as its box name the function name of the function represented by the box.

3.417

box number

- single digit (0, 1, 2, ..., 9) placed in the lower right corner of a box to uniquely identify that box in a diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.20*]

Note 1 to entry: The only box that can be numbered 0 is the box that represents the A0 function in A-0 and A-1 context diagrams.

3.418

BP

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

3.419

brainstorming

1. a general data gathering and creativity technique that can be used to identify risks, ideas, or solutions to issues by using a group of team members or subject-matter experts [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.420

branch

1. computer program construct in which one of two or more alternative sets of program statements is selected for execution **2.** point in a computer program at which one of two or more alternative sets of program statements is selected for execution **3.** junction at which a root arrow segment (going from source to use) divides into two or more arrow segments [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.21*] **4.** to perform the selection in (1) **5.** any of the alternative sets of program statements in (1) **6.** set of evolving source file versions [*ISO/IEC TR 19759:2016, Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK), 6.1.3*] **7.** deviation from the main development line for a configuration item, which allows different persons to work on the same item at the same time [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities, 3.3*]
cf. case, jump, go to, if-then-else

Note 1 to entry: Every branch is identified by a tag. Often, a branch identifies the file versions that have been or will be released as a product release. It can denote unbundling of arrow meaning, i.e., the separation of object types from an object type set. Also refers to an arrow segment into which a root arrow segment has been divided.

3.421

branch testing

1. testing designed to execute each outcome of each decision point in a computer program
cf. path testing, statement testing

3.422

breadcrumb trail

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

3.423

break-even analysis

1. analysis of two or more objective functions to find where, if at all, they have the same value

3.424

breakpoint

1. point in a computer program at which execution can be suspended to permit manual or automated monitoring of program performance or results

Note 1 to entry: Types include code breakpoint, data breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint. A breakpoint is said to be set when both a point in the program and an event that will cause suspension of execution at that point are defined; it is said to be initiated when program execution is suspended.

3.425

browser

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

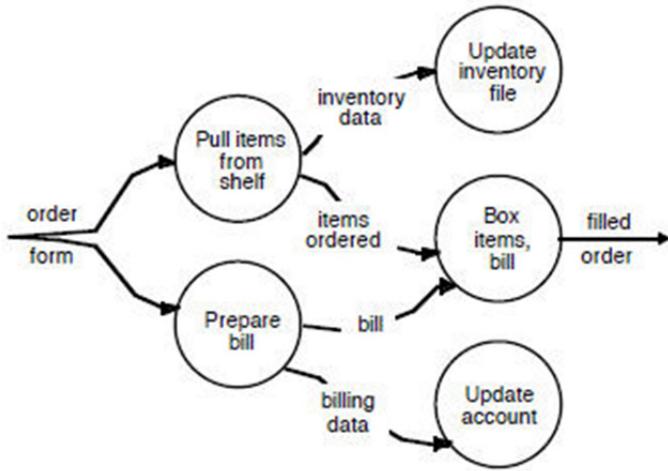
3.426

BRS

1. business requirements specification [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.2*]

3.427**bubble chart**

- 1.** data flow, data structure, or other diagram in which entities are depicted with circles (bubbles) and relationships are represented by links drawn between the circles
cf. block diagram, box diagram, flowchart, graph, input-process-output chart, structure chart

**Figure 5 —Bubble chart****3.428****budget**

- 1.** the approved estimate for the project or any work breakdown structure component or any schedule activity [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. estimate

Note 1 to entry: often used also to refer to work effort as well as, or instead of, money.

3.429**budget at completion (BAC)**

- 1.** the sum of all the budgets established for the work to be performed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.430**buffer**

- 1.** device or storage area used to store data temporarily to compensate for differences in rates of data flow, time of occurrence of events, or amounts of data that can be handled by the devices or processes involved in the transfer or use of the data **2.** routine that accomplishes the objectives in (1) **3.** to allocate, schedule, or use devices or storage areas as in (1)

3.431**build**

- 1.** operational version of a system or component that incorporates a specified subset of the capabilities that the final product will provide [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1; ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*] **2.** process of generating (archiving) an executable and testable system from source versions or baselines [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1; ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities, 3.4*] **3.** to perform the steps required to produce an instance of the product [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]

Note 1 to entry: In software, this means processing source files to derive target files. In hardware, this means assembling a physical object. The build needs to compile and link the various versions in the correct order. The build tools can be integrated into a configuration management tool.

3.432

build process

1. process of transforming project code base into usable applications [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.433

built-in class

1. class that is a primitive in the IDEF1X metamodel [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.13*]

3.434

built-in random access memory (RAM)

1. RAM embedded in a microcontroller unit (MCU) chip

3.435

built-in read only memory

built-in ROM

1. read-only memory embedded in a microcontroller unit (MCU) chip

3.436

bundle

1. grouping of products which is the result of a marketing/licensing strategy to sell entitlements to multiple products as one purchased item [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.5*] **2.** arrow segment that collects multiple meanings into a single construct or abstraction, i.e., an arrow segment that represents an object type set that includes more than one object type [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.22*] **3.** to combine separate arrow meanings into a composite arrow meaning, expressed by joining arrow segments, i.e., the inclusion of multiple object types into an object type set [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.22*]

Note 1 to entry: A bundle can be referred to as a suite, if the products are closely related and typically integrated (such as an office suite containing a spreadsheet, word processor, presentation and other related items). Bundles can also refer to software titles that are less closely related such as a game, a virus scanner and a utility bundled together with a new computer, or to groups of entitlements, such as multiple entitlements for a backup software product.

3.437

burndown

1. an indicator of the work completed and an estimate of remaining work to be completed or remaining effort needed to complete a product development iteration cycle. Work is measured as all work done to deliver story points, stories, features, functions, function points, user stories, use cases, or requirements during a product development iteration. [*Software Extension to the PMBOK® Guide Fifth Edition*]

cf. burnup

3.438

burndown chart

1. document that records project status, usually showing tasks completed against total number of tasks [*ISO/IEC/IEEE 26515: 2011 Systems and software engineering: Developing user documentation in an agile environment, 4.4*]

3.439

burndown rate

1. the number of software story points, features, functions, user stories, use cases, or requirements completed per work unit (week or iteration) [*Software Extension to the PMBOK® Guide Fifth Edition*]

cf. velocity

3.440

burnup

1. an indicator of the number of story points, features, functions, user stories, use cases, or requirements completed and the work remaining or remaining effort needed to complete a product development iteration cycle.

Work is measured as all work done to deliver story points, features, functions, user stories, use cases, or requirements during a product development iteration. [Software Extension to the PMBOK® Guide Fifth Edition] cf. burndown

3.441

bus

- 1.** data communication path in a computer or system

3.442

business case

- 1.** a documented economic feasibility study used to establish validity of the benefits of a selected component lacking sufficient definition and that is used as a basis for the authorization of further project management activities [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.443

business information management

- 1.** domain responsible for all of the tasks and activities that are aimed at supporting the end users in the use of the application and at acting as the customer of the IT organizations [ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.8]

Note 1 to entry: Business information management represents the business as the customer organization or client of the application management and IT infrastructure management organizations in maintaining the functionality of the information provisioning and the information systems. It is the demand side of the information provisioning. An information system can have non-automated elements such as forms and user guides. Those elements are usually maintained by the business information management organization.

3.444

business objective

- 1.** strategy designed by senior management to ensure an organization's continued existence and enhance its profitability, market share, and other factors influencing the organization's success

3.445

business process

- 1.** partially ordered set of enterprise activities that can be executed to achieve some desired end-result in pursuit of a given objective of an organization

3.446

business value

- 1.** a concept that is unique to each organization and includes tangible and intangible elements. Through the effective use of project, program, and portfolio management disciplines, organizations will possess the ability to employ reliable, established processes to meet enterprise objectives and obtain greater business value from their investments. [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.447

busy

- 1.** pertaining to a system or component that is operational, in service, and in use
cf. down, idle, up

3.448

busy time

- 1.** in computer performance engineering, the period of time during which a system or component is operational, in service, and in use
cf. down time, idle time, set-up time, up time

Note 1 to entry: operational, in service, and in use

3.449

buyer

- 1.** acquirer of products, services, or results for an organization [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]
cf. customer, acquirer

3.450

byte

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

3.451

C4I

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

3.452

cache

1. temporary storage in computer memory, to improve operations by having frequently used data readily available for retrieval **2.** RAM with very high operating speed used for data storage within a processor

3.453

CAD

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

3.454

cadence

1. frequency of performing a periodic activity, such as incremental product release [Software Extension to the PMBOK® Guide Fifth Edition]

3.455

CAI

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

3.456

calculator

1. device that is suitable for performing arithmetic operations, but that requires human intervention to alter its stored program, if any, and to initiate each operation or sequence of operations [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

Note 1 to entry: A calculator performs some of the functions of a computer, but usually operates only with frequent human intervention.

3.457

calendar unit

1. smallest unit of time used in scheduling a project

Note 1 to entry: Calendar units are generally in hours, days, or weeks, but can also be in quarter years, months, shifts, or even in minutes.

3.458

call

1. transfer of control from one software module to another, usually with the implication that control will be returned to the calling module **2.** computer instruction that transfers control from one software module to another as in (1) and often specifies the parameters to be passed to and from the module **3.** to transfer control from one software module to another as in (1) and, often, to pass parameters to the other module **4.** request for service(s) or action(s) with respect to an application or a related service [ISO/IEC 16350-2015 *Information technology — Systems and software engineering — Application management*, 4.9]
cf. go to

Note 1 to entry: A call might concern a request for service, information or advice; disruption or error reporting (incident); request for change; assignment (for instance an instruction to start an off-schedule production run); and complaint.

3.459

call arrow

- 1.** arrow that enables the sharing of detail between IDEF0 models (linking them together) or within an IDEF0 model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.23*]

Note 1 to entry: The tail of a call arrow is attached to the bottom side of a box. One or more page references are attached to a call arrow.

3.460

call by name

- 1.** method for passing parameters, in which the calling module provides to the called module a symbolic expression representing the parameter to be passed, and a service routine evaluates the expression and provides the resulting value to the called module

cf. call by reference, call by value

Note 1 to entry: Because the expression is evaluated each time its corresponding formal parameter is used in the called module, the value of the parameter can change during the execution of the called module.

3.461

call by reference

call by address

call by location

- 1.** a method for passing parameters, in which the calling module provides to the called module the address of the parameter to be passed

cf. call by name, call by value

Note 1 to entry: With this method, the called module has the ability to change the value of the parameter stored by the calling module.

3.462

call by value

- 1.** method of passing parameters, in which the calling module provides to the called module the actual value of the parameter to be passed

cf. call by name, call by reference

Note 1 to entry: With this method, the called module cannot change the value of the parameter as stored by the calling module.

3.463

call graph

call tree,

tier chart

- 1.** diagram that identifies the modules in a system or computer program and shows which modules call one another

cf. structure chart, control flow diagram, data flow diagram, data structure diagram, state diagram

Note 1 to entry: The result is not necessarily the same as that shown in a structure chart.

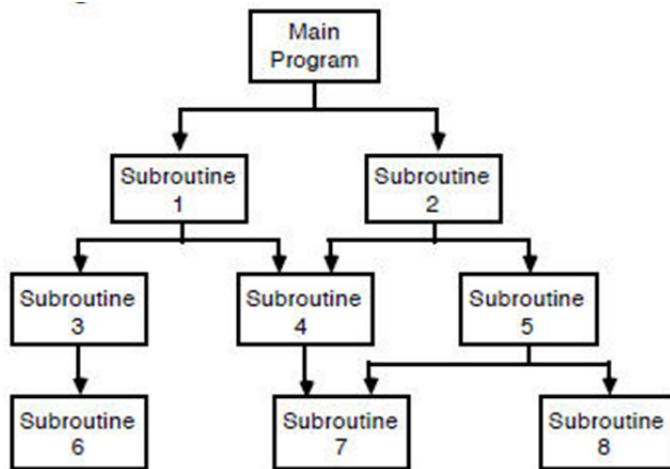


Figure 6 —Call graph

3.464**call list**

1. ordered list of arguments used in a call to a software module

3.465**call reference**

1. page reference attached to a call arrow [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.24*]

3.466**called diagram**

1. decomposition diagram invoked by a calling box and identified by a page reference attached to a call arrow [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.25*]

3.467**calling box**

1. box that is detailed by a decomposition diagram that is not the box's child diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.26*]

Note 1 to entry: A call arrow is attached to the bottom of a calling box.

3.468**calling sequence**

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

3.469**CAN**

1. controller area network

3.470**candidate FSM method**

1. documented software size measurement method submitted for conformity evaluation [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.1*]

3.471**candidate key**

1. attribute, or combination of attributes, of an entity for which no two instances agree on the values [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.14*]

Note 1 to entry: [key style]

3.472

capability maturity model

1. model that contains the essential elements of effective processes for one or more disciplines and describes an evolutionary improvement path from ad hoc, immature processes to disciplined, mature processes with improved quality and effectiveness

3.473

capable process

1. process that can satisfy specified product quality, service quality, and process-performance objectives
cf. stable process, standard process, statistically managed process

3.474

capacity

1. degree to which the maximum limits of a product or system parameter meet requirements [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.2.3*]

Note 1 to entry: Parameters can include the number of items that can be stored, the number of concurrent users, the communication bandwidth, throughput of transactions, and size of database.

3.475

capacity testing

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

3.476

capital expenditure

1. spending by an enterprise to acquire tangible infrastructure or facilities items, such as furniture, computers, and the like

Note 1 to entry: does not include acquisition of consumable supplies or of items to be included in finished products for sale

3.477

capsule

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

EXAMPLE: a virtual machine (e.g. a process)

3.478

capsule manager

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

3.479

CARD

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

3.480

cardinality

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

Note 1 to entry: For each direction of a relationship, the cardinality can be constrained.

3.481

cardinality constraint

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

3.482

carrier

1. set of a many-sorted algebra [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.6]

3.483

case

multiple exclusive selective construct

1. single-entry, single-exit multiple-way branch that defines a control expression, specifies the processing to be performed for each value of the control expression, and returns control in all instances to the statement immediately following the overall construct
 2. Computer Aided Software Engineering [*ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools*, 4]
- cf. go to, jump, if-then-else. multiple inclusive selective construct*

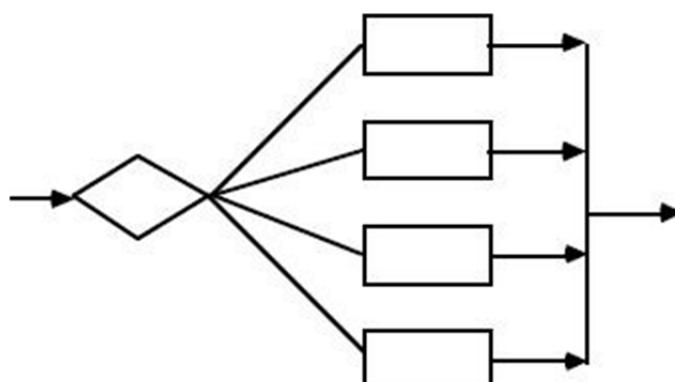


Figure 7 —Case construct

3.484

CASE needs

1. organizational requirements which are met by CASE tool characteristics [*ISO/IEC TR 14471:2007 Information technology — Software engineering — Guidelines for the adoption of CASE tools*, 2.1.3]

Note 1 to entry: These characteristics are detailed in ISO/IEC 14102:2008. They include management process, development process, maintenance, documentation, configuration management, quality assurance, verification, validation, environment needs, CASE tool integrability, quality characteristics, acquisition needs, implementation needs, support indicators, and certification requirements.

3.485

CASE tool

1. software tool used for computer-aided software engineering (CASE) [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description*, 3.3]
2. software product that can assist software engineers by providing automated support for software life-cycle activities [*ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools*, 3.2]
3. software product that can assist software and system engineers by providing automated support for software and system engineering life-cycle activities [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services*, 2.3]

Note 1 to entry: A CASE tool can provide support in only selected functional areas or in a wide variety of functional areas.

3.486

cast

1. to treat an object of one type as an object of another type [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.17*]
cf. coerce

3.487

catastrophic failure

1. failure of critical software

3.488

categorization

1. specific way to allocate a target system into a category [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it, 3.5*]
cf. generalization

3.489

categorization scheme

1. orderly combination of views and categories related to software.

3.490

categorization space

1. universal set of systems and software which has one or more classification axes as its individual dimension, by which stakeholder's concerns on categorization are expressed [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it, 3.6*]

3.491

category

1. specifically defined division or grouping of software based upon one or more attributes or characteristics 2. subset of categorization space, which the stakeholders are interested in, specified using a combination of one or more equivalence classes [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it, 3.9*]

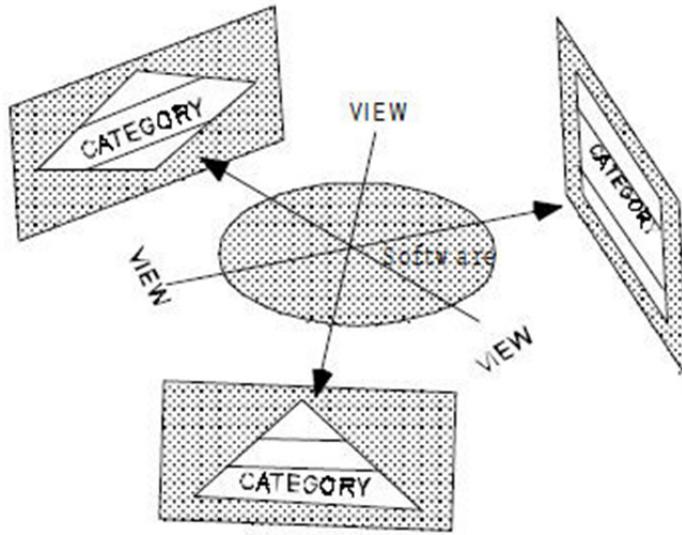


Figure 8 — Category

3.492

category entity

1. entity whose instances represent a subtype or subclassification of another entity (generic entity) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.21*]
cf. subclass, subtype

Note 1 to entry: [key style]

3.493

causal analysis

- 1.** analysis of a defect to determine its cause

3.494

causal relationship

- 1.** existence dependency between the stimulus of a behavior and the various responses that are caused to occur [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.9*]

3.495

cause and effect diagram

fishbone diagram

- 1.** a decomposition technique that helps trace an undesirable effect back to its root cause [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.496

caution

- 1.** advisory in software user documentation that performing some action can lead to consequences that are unwanted or undefined, such as loss of data or an equipment problem [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.7*]

cf. warning, note

3.497

CBA

- 1.** conduct benchmarking activity [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking, 4*]

3.498

CCB

- 1.** configuration control board [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2; ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*] **2.** change control board [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2; ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*]

3.499

CCCS

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

3.500

CCM

- 1.** CORBA Component Model [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.501

CCS

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

3.502

CD

- 1.** Compact Disk

3.503

CD-ROM

- 1.** compact disc, read-only memory [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2*]

3.504

CDD

1. capability development document [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.505

CDIF

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

3.506

CDIF clear text encoding

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

3.507

CDIF exporter

1. tool that creates a CDIF transfer file [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

3.508

CDIF family of standards

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

3.509

CDIF graphical notation

1. set of rules governing the representation of CDIF modeling concepts in diagrams [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

3.510

CDIF identifier

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

3.511

CDIF importer

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

3.512

CDIF meta-metamodel

1. description of the set of concepts and notations used to define a metamodel [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Specifically, the CDIF meta-metamodel defines an Entity-Relationship-Attribute model that is used to construct and define both metamodels and the CDIF meta-metamodel itself.

3.513

CDIF metaidentifier

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

3.514

CDIF semantic metamodel

1. description of the set of concepts and notations used to define a model [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: The CDIF semantic metamodel defines an Entity-Relationship-Attribute model that is used to construct and define models used in systems development.

3.515

CDIF transfer

- 1.** combination of a particular syntax, a particular encoding of that syntax, and a metamodel [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: In other words, a complete definition of the format and contents of a transfer.

3.516

CDIF transfer file

- 1.** transfer file conforming to ISO/IEC 15475 [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.1]

3.517

CDIF transfer format

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

3.518

CDIF transfer syntax and encoding

- 1.** standard vehicle format supported by CDIF [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: The combination of SYNTAX.1 and ENCODING.1 forms the initial CDIF transfer syntax and encoding.

3.519

CDR

- 1.** critical design review [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2] **2.** common data representation [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.3]

3.520

CDRL

- 1.** contract data requirements list [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 3]

3.521

central processing unit (CPU)

- 1.** functional unit that consists of one or more processors and their internal storage [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.522

central tendency

- 1.** a property of the central limit theorem predicting that the data observations in a distribution will tend to group around a central location. The three typical measures of central tendency are the mean, median, and mode [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.523

certification

- 1.** third-party attestation related to products, processes, systems, or persons [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.15] **1.** a written guarantee that a system or component complies with its specified requirements and is acceptable for operational use **2.** formal demonstration that a system or component complies with its specified requirements and is acceptable for operational use **3.** process of confirming that a system or component complies with its specified requirements and is acceptable for operational use

EXAMPLE: written authorization that a computer system is secure and is permitted to operate in a defined environment

Note 1 to entry: Certification is applicable to all objects of conformity assessment except for conformity assessment bodies themselves, to which accreditation is applicable. Certification of a management system is sometimes also called registration.

3.524

certification artifact

- 1.** tangible results from a certification process

EXAMPLE: inspection checklists, metrics, problem reports

3.525

certification body

- 1.** body certifying persons against the requirements in ISO/IEC 24773:2008, including the development and maintenance of a Scheme [*ISO/IEC 24773:2008 Software engineering -Certification of software engineering professionals -Comparison framework*, 3.2] **2.** third-party conformity assessment body operating certification schemes [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.16]

Note 1 to entry: A certification body can be non-governmental or governmental (with or without regulatory authority).

3.526

certification criteria

- 1.** set of standards, rules, or properties to which an asset must conform in order to be certified to a certain level

Note 1 to entry: Certification criteria are defined by a certification policy. Certification criteria can be specified as a set of certification properties that must be met.

3.527

certification process

- 1.** process of assessing whether an asset conforms to predetermined certification criteria appropriate for that class of asset **2.** activities by which a certification body establishes that a person fulfills specified competence requirements, including application, evaluation, decision on certification, surveillance and recertification, and use of certificates and logos or marks [*ISO/IEC TR 29154:2013, Software engineering — Guide for the application of ISO/IEC 24773:2008 (Certification of software engineering professionals — Comparison framework)*, 3.1]

3.528

certification property

- 1.** statement about some feature or characteristic of an asset that can be assessed as being true or false during a certification process

Note 1 to entry: Properties can relate to what an asset is, what it does, or how it relates to its operating environment. An assessment of a certification quality factor is accomplished by assessing the underlying certification properties.

3.529

certification scheme

- 1.** specific certification requirements related to specified categories of persons, to which the same particular standards and rules and the same procedures apply [*ISO/IEC TR 29154:2013, Software engineering — Guide for the application of ISO/IEC 24773:2008 (Certification of software engineering professionals — Comparison framework)*, 3.2] **2.** certification system related to specified products, to which the same specified requirements, specific rules, and procedures apply [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.17]

Note 1 to entry: [ISO/IEC 17024]

3.530

certification scheme owner

- 1.** person or organization that is responsible for developing and maintaining a specific certification scheme [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.18]

Note 1 to entry: The certification scheme owner can be the certification body itself, a governmental authority, trade association, group of certification bodies, or other.

3.531

CFD

- 1.** cumulative flow diagram [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.532

CFP

1. COSMIC function point [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 3*]

3.533

chain

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

3.534

chain type

1. classification of chains which is defined by the sequence of tasks types [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.4*]

Note 1 to entry: The emulated users submit only chains of specified chain types to the SUT.

3.535

change

1. the modification of an existing application comprising additions, changes and deletions [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]
cf. enhancement

3.536

change authority

1. configuration board [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management, 4.2*]

Note 1 to entry: Disposition is made by a designated change authority traditionally given the name "Change/Configuration Control Board". This authority can approve a proposed change, thus converting it to an approved modification, or can disapprove a proposed change, or can defer a decision.

3.537

change control

1. a process whereby modifications to documents, deliverables, or baselines associated with the project are identified, documented, approved, or rejected [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. configuration control, version control

3.538

change control board (CCB)

1. a formally chartered group responsible for reviewing, evaluating, approving, delaying, or rejecting changes to a project, and for recording and communicating such decisions [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. configuration control board

3.539

change control procedure

1. actions taken to identify, document, review, and authorize changes to a software or documentation product that is being developed [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.8*]

Note 1 to entry: The procedures ensure that the validity of changes is confirmed, that the effects on other items are examined, and that those people concerned with the development are notified of the changes.

3.540

change control system

1. a set of procedures that describes how modifications to the project deliverables and documentation are managed and controlled [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.541

change dump

differential dump

1. selective dump of those storage locations whose contents have changed since some specified time or event
cf. dynamic dump, memory dump, postmortem dump, selective dump, snapshot dump, static dump

3.542

change log

1. a comprehensive list of changes made during the project. This typically includes dates of the change and impacts in terms of time, cost, and risk [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.543

change management

1. judicious use of means to effect a change, or a proposed change, to a product or service
cf. configuration management

3.544

change package

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

3.545

change project function point count

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

3.546

change record

1. record containing details of which configuration items are affected and how they are affected by an authorized change

3.547

change request

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

3.548

change set

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

3.549

changeover system

1. temporary information processing system used to facilitate the transition from an operational system to its successor [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.550

channel

distribution channel

1. approach to distributing products and services from the original supplier to the end-user organization [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.3*] 2. configuration of stubs, binders, protocol objects and interceptors providing a binding between a set of interfaces to basic engineering objects, through which interaction can occur [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.8*]

Note 1 to entry: Typical channels for software include direct, VAR, OEM, reseller, and educational reseller. Bindings that require channels are referred to as distributed bindings in the engineering language; bindings between engineering objects that do not require channels (e.g. between engineering objects in the same cluster) are referred to as local bindings.

3.551

channel capacity

1. maximum amount of information that can be transferred on a given channel per unit of time; usually measured in bits per second or in baud.

cf. memory capacity, storage capacity

3.552

channel partner

1. person or entity working with a software licensor or another person/entity within the channel who facilitates the sale of software to the end-user [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.4*]

EXAMPLE: original equipment manufacturer (OEM), reseller, vendor

3.553

character

1. letter, digit, or other symbol that is used to represent information **2.** member of a set of elements that is used for the representation, organization, or control of data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.554

character set

1. collection of characters used in an encoding to represent terminal symbols [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: The character set used is significant in the encoding of text and string meta-attributes for a CDIF transfer.

3.555

character type

1. data type whose members can assume the values of specified characters and can be operated on by character operators, such as concatenation

cf. enumeration type, integer type, logical type, real type

3.556

characteristic entity

1. meta-entity that provides additional attribution for another meta-object [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Other common names for characteristic entity are attributive entity and dependent entity. Each instance of a characteristic meta-entity is logically only related to one instance of one other meta-object, therefore an importer could incorporate the meta-attributes of a characteristic meta-entity with those of the 'owning' meta-object, where the owning meta-object is the one to which the characteristic meta-entity is related with a cardinality of 1:1.

3.557

characteristic of FUR

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

3.558

chart of accounts

1. numbering system used by a project or organization to identify costs by category, such as labor, supplies, materials, and equipment

cf. code of accounts

3.559

checklist analysis

1. a technique for systematically reviewing materials using a list for accuracy and completeness [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.560

checkout

- 1.** testing conducted in the operational or support environment to ensure that a software product performs as required after installation.

3.561

checkpoint

- 1.** point in a computer program at which program state, status, or results are checked or recorded **2.** object template derived from the state and structure of an engineering object that can be used to instantiate another engineering object, consistent with the state of the original object at the time of checkpointing [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.20]

3.562

checkpointing

- 1.** creating a checkpoint [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.21]

Note 1 to entry: Checkpoints can only be created when the engineering object involved satisfies a pre-condition stated in a checkpointing policy

3.563

checksheets

- 1.** a tally sheet that can be used as a checklist when gathering data [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.564

chief programmer

- 1.** leader of a chief programmer team; a senior-level programmer whose responsibilities include producing key portions of the software assigned to the team, coordinating the activities of the team, reviewing the work of the other team members, and having an overall technical understanding of the software being developed
cf. backup programmer, chief programmer team

3.565

chief programmer team

- 1.** software development group that consists of a chief programmer, a backup programmer, a secretary/librarian, and additional programmers and specialists as needed, and that employs procedures designed to enhance group communication and to make optimum use of each member's skills
cf. backup programmer, chief programmer, egoless programming

3.566

child box

- 1.** box in a child diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.27]

3.567

child diagram

- 1.** decomposition diagram related to a specific box by exactly one child/parent relationship [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.28]

3.568

child entity

- 1.** entity in a specific relationship whose instances can be related to zero or one instance of the other entity (parent entity) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.22]

Note 1 to entry: [key style]

3.569

child tag

- 1.** tag that has a subsidiary relationship to another tag

EXAMPLE: child entitlement tags created for allocation purposes

3.570

CI

1. configuration item

3.571

CIDL

1. Component Implementation Definition Language [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.572

CIF

1. Component Implementation Framework [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.573

CIM

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

3.574

clabject

1. dual entity that is a class and an object at the same time [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 3.13]

Note 1 to entry: Because of their dual nature, clabjects exhibit a class facet and an object facet, and can work as either at any time. Instances of powertypes are usually viewed as clabjects, since they are objects (because they are instances of a type, the powertype) and also classes (subtypes of the partitioned type).

3.575

claim

1. a request, demand, or assertion of rights by a seller against a buyer, or vice versa, for consideration, compensation, or payment under the terms of a legally binding contract, such as for a disputed change [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** true-false statement about the limitations on the values of an unambiguously defined property — called the claim's property — and limitations on the uncertainty of the property's values falling within these limitations during the claim's duration of applicability under stated conditions [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.1.2] **3.** proposition representing a requirement of the system-of-interest that enables the system-of-interest to achieve tolerable risk if it were met [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.2]

Note 1 to entry: Claims usually relate to specified versions of a product. The statement of a claim does not mean that the only possible intent or desire is to show it is true. Sometimes claims are made for the purpose of evaluating whether they are true or false or undertaking an effort to establish what is true. In its entirety, a claim is an unambiguous declaration of an assertion with any associated conditionality, giving explicit details including limitations on values and uncertainty. It could be about the future, present, or past. A safety goal is an instance of a claim.

3.576

claims administration

1. the process of processing, adjudicating, and communicating contract claims [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.577

class

1. abstraction of the knowledge and behavior of a set of similar things [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.23] **2.** static programming entity in an object-oriented program that contains a combination of functionality and data3. of <X>s, the set of all <X>s satisfying a type [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.10]

cf. type

Note 1 to entry: Classes are used to represent the notion of "things whose knowledge or actions are relevant."

3.578

class hierarchy

- 1.** ordering of classes, in which a subclass is a specialization of its superclass

Note 1 to entry: A class inherits attributes and relationships from its superclass and can define additional attributes and relationships of its own.

3.579

class-level attribute

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

3.580

class-level operation

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

3.581

class-level responsibility

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

cf. instance-level responsibility

EXAMPLE: The total registeredVoterCount would be a class-level property of the class registeredVoter; there would be only one value of registeredVoterCount for the class as a whole.

3.582

classification

- 1.** manner in which the assets are organized for ease of search and extraction within a reuse library [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]

3.583

classification axis

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

3.584

clear

- 1.** to set a variable, register, or other storage location to zero, blank, or other null value

cf. initialize, reset

3.585

clear text file encoding

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

3.586

client

- 1.** code or process that invokes an operation on an object [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 3.2.3*] **2.** of a service, any entity capable of requesting the service [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces, 5.3*] **3.** for certification, the organization that is responsible to a certification body for ensuring certification requirements, including product requirements, are fulfilled [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.19*]

3.587

client object

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

3.588

client-side

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

3.589

clock pulse generator (CPG)

1. electronic unit to produce uniform timed signals

EXAMPLE: embedded in an MCU

3.590

cloning

1. instantiating a cluster from a cluster checkpoint [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.24]

3.591

Close Procurements

1. the process of completing each project procurement [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.592

Close Project or Phase

1. the process of finalizing all activities across all of the project management process groups to formally complete a project or phase [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.593

closed loop

1. loop that has no exit and whose execution can be interrupted only by intervention from outside the computer program or procedure in which the loop is located

cf. UNTIL, WHILE

3.594

closed procurements

1. project contracts or other procurement agreements that have been formally acknowledged by the proper authorizing agent as being finalized and signed off [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.595

closed subroutine

1. subroutine that is stored at one given location rather than being copied into a computer program at each place that it is called

cf. open subroutine

3.596

closed term

ground term

1. term comprising constants and operators but no variables [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.24.1]

3.597

closing process group

1. those processes performed to finalize all activities across all Process Groups to formally close a project or phase [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.598

cluster

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

EXAMPLE: a segment of virtual memory containing objects

3.599

cluster checkpoint

1. cluster template containing checkpoints of the basic engineering objects in a cluster [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.22*]

3.600

cluster manager

1. engineering object which manages the basic engineering objects in a cluster [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.3*]

3.601

cluster template

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

3.602

CM

1. configuration management

3.603

CM service

configuration management service

1. abstract description of work done by CM tools [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities, 3.8*]

3.604

CM tool

configuration management tool

1. software product that can assist software engineers by providing automated support for configuration management activities [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities, 3.9*]

3.605

CMDB

1. configuration management database [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2*]

3.606

CMIP

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

3.607

CMIR

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

3.608

CMIS

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

3.609

CMP

1. configuration management plan [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2*]

3.610

CMS

1. configuration management system [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2*]

3.611

CMT

1. container-managed transaction [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.612

CNCS

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

3.613

co-existence

1. degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.3.1*]

3.614

code

1. in software engineering, computer instructions and data definitions expressed in a programming language or in a form output by an assembler, compiler, or other translator **2.** to express a computer program in a programming language **3.** character or bit pattern that is assigned a particular meaning
cf. source code, object code, machine code, micro code

EXAMPLE: a status code

3.615

code breakpoint

control breakpoint

1. breakpoint that is initiated upon execution of a given computer instruction
cf. data breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint

3.616

code freeze

1. period during which non-critical changes to the code are not allowed

3.617

code generator

1. software tool that accepts as input the requirements or design for a computer program and produces source code that implements the requirements or design generator
cf. application

3.618

code of accounts

1. a numbering system used to uniquely identify each component of the work breakdown structure (WBS) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. chart of accounts

3.619

code of ethics standard

- 1.** standard that describes the characteristics of a set of moral principles dealing with accepted standards of conduct by, within, and among professionals

3.620

code review

- 1.** meeting at which software code is presented to project personnel, managers, users, customers, or other interested parties for comment or approval

cf. design review, formal qualification review, requirements review, test readiness review

3.621

code tuning

- 1.** process of making statement-level changes to a program to make it more efficient **2.** changes made to program source code for the purpose of optimizing performance, usually to increase speed or reduce memory usage

3.622

coding

- 1.** in software engineering, the process of expressing a computer program in a programming language **2.** transforming of logic and data from design specifications (design descriptions) into a programming language

3.623

coerce

- 1.** to treat an object of one type as an object of another type by using a different object [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.28*] *cf.* cast

3.624

cognitive level

- 1.** qualitative assessment of an individual's familiarity with a given topic [*ISO/IEC 24773:2008 Software engineering -Certification of software engineering professionals -Comparison framework, 3.3*]

3.625

cohesion

module strength

- 1.** manner and degree to which the tasks performed by a single software module are related to one another **2.** in software design, a measure of the strength of association of the elements within a module [*ISO/IEC TR 19759:2016, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOk)*], 2.1.4 *cf.* coupling

Note 1 to entry: Types include coincidental, communicational, functional, logical, procedural, sequential, and temporal.

3.626

coincidental cohesion

- 1.** type of cohesion in which the tasks performed by a software module have no functional relationship to one another

cf. communicational cohesion, functional cohesion, logical cohesion, procedural cohesion, sequential cohesion, temporal cohesion

3.627

collaboration

- 1.** cooperative exchange of requests among classes and instances in order to achieve some goal [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.29*]

3.628

collapse

- 1.** to terminate development on one branch by integrating it with another

3.629

Collect Requirements

1. the process of determining, documenting, and managing stakeholder needs and requirements to meet project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.630

collection

1. unrestricted grouping of software behavior concept instances into a particular named subset [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.631

collection cardinality

1. specification, for a collection-valued property, of how many members the value of the property, that is, the collection, can or is required to have for each instance [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.30*]
cf. cardinality constraint

3.632

collection class

1. class in which each instance is a group of instances of other classes [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.31*]

3.633

collection-valued

1. value that is complex [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.33*]
cf. scalar

Note 1 to entry: That is, having constituent parts.

3.634

collection-valued class

1. class in which each instance is a collection of values [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.34*]
cf. scalar-valued class

3.635

collection-valued property

collection property

1. property that maps to a collection class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.35*]
cf. scalar-valued property

3.636

colocation

1. an organizational placement strategy where the project team members are physically located close to one another in order to improve communication, working relationships, and productivity [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.637

comfort

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

3.638

command

1. expression that can be input to a computer system to initiate an action or affect the execution of a computer program

EXAMPLE: the 'logon' command to initiate a computer session

3.639

command language

- 1.** language used to express commands to a computer system

cf. command-driven

3.640

command-driven

- 1.** pertaining to a system or mode of operation in which the user directs the system through commands

cf. menu-driven

3.641

comment

- 1.** information embedded within a computer program, job control statements, or a set of data that provides clarification to human readers but does not affect machine interpretation

3.642

commercial-off-the-shelf (COTS)

- 1.** [software] product available for purchase and use without the need to conduct development activities [*ISO/IEC 90003:2014 Software engineering — Guidelines for the application of ISO 9001:2008 to computer software*, 3.4; *ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.4]

cf. software product

Note 1 to entry: COTS software product includes the product description (including all cover information, data sheet, web site information, etc.), the user documentation (necessary to install and use the software), the software contained on a computer sensible media (disk, CD-ROM, internet downloadable, etc.). Software is mainly composed of programs and data. This definition applies also to product descriptions, user documentation and software which are produced and supported as separate manufactured goods, but for which typical commercial fees and licensing considerations do not apply.

3.643

commercial-off-the-shelf software product

COTS software product

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

3.644

commit

- 1.** to integrate the changes made to a developer's private view of the source code into a branch accessible through the version control system's repository

3.645

commit message

- 1.** explanatory message accompanying a commit

Note 1 to entry: often contains a brief description of the change and its rationale; names of contributors, reviewers, or approvers; a reference to third-party software from which the change was obtained; a schedule for integrating it to other branches; and a reference to the issue identifier associated with the change

3.646

commit privileges

- 1.** person's authority to commit changes

Note 1 to entry: Sometimes privileges are associated with a specific part of the product (for example, artwork or documentation) or a specific branch.

3.647

commit war

- 1.** series of conflicting and mutually reversing commits introduced by developers who disagree on how a particular element is being coded

Note 1 to entry: sometimes starts with a hostile backout.

3.648

commit window

- 1.** period during which commits are allowed for a specific branch

Note 1 to entry: In some development environments, commit windows for a maintenance branch might only open for short periods a few times a year.

3.649

commitment

- 1.** action resulting in an obligation by one or more of the participants in the act to comply with a rule or perform a contract [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.6.2*]

Note 1 to entry: The enterprise object(s) participating in an action of commitment can be parties or agents acting on behalf of a party or parties. In the case of an action of commitment by an agent, the principal becomes obligated.

3.650

committer

- 1.** developer with commit privileges

3.651

common ancestor constraint

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

3.652

common cause

random cause

- 1.** source of variation of a process that exists because of normal and expected interactions among components of a process

cf. special cause

Note 1 to entry: On a control chart, it appears as part of the random process variation (i.e., variation from a process that would be considered normal or not unusual), and is indicated by a random pattern of points within the control limits.

3.653

common storage

common area

common block

- 1.** portion of main storage that can be accessed by two or more modules in a software system

cf. global data

3.654

common-environment coupling

common coupling

- 1.** type of coupling in which two software modules access a common data area

cf. content coupling, control coupling, data coupling, hybrid coupling, pathological coupling

3.655

commonality

- 1.** set of functional and non-functional characteristics that is shared by all applications belonging to the product line [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.9*]

3.656

communication constraints

- 1.** restrictions on the content, timing, audience, or individual who will deliver a communication usually stemming from specific legislation or regulation, technology, or organizational policies [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.657

communication interface

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

3.658

communication management

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

3.659

communication management plan

1. a component of the project, program, or portfolio management plan that describes how, when, and by whom information about the project will be administered and disseminated [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.660

communication methods

1. a systematic procedure, technique, or process used to transfer information among project stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.661

communication models

1. a description, analogy or schematic used to represent how the communication process will be performed for the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.662

communication requirements analysis

1. an analytical technique to determine the information needs of the project stakeholders through interviews, workshops, study of lessons learned from previous projects, etc. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.663

communication technology (CT)

1. specific tools, systems, computer programs, etc., used to transfer information among project stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.664

communicational cohesion

1. type of cohesion in which the tasks performed by a software module use the same input data or contribute to producing the same output data

cf. coincidental cohesion, functional cohesion, logical cohesion, procedural cohesion, sequential cohesion, temporal cohesion

3.665

communications domain

1. set of protocol objects capable of interworking [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.13*]

3.666

communications planning

1. process of defining how to meet the information and communication needs of the stakeholders: who needs what information, when they need it, and how it will be given to them

3.667

community

1. configuration of objects formed to meet an objective [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 5.1.1*]

Note 1 to entry: The objective is expressed as a contract which specifies how the objective can be met.

3.668

community object

- 1.** composite enterprise object that represents a community [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.2.2*]

Note 1 to entry: Components of a community object are objects of the community represented

3.669

compact disc read only memory (CD-ROM)

- 1.** optical disk which can be read, but not erased or rewritten

3.670

compaction

- 1.** in microprogramming, the process of converting a microprogram into a functionally equivalent microprogram that is faster or shorter than the original
cf. local compaction, global compaction

3.671

comparator

- 1.** software tool that compares two computer programs, files, or sets of data to identify commonalities or differences

Note 1 to entry: Typical objects of comparison are similar versions of source code, object code, database files, or test results.

3.672

compatibility

- 1.** degree to which a product, system or component can exchange information with other products, systems or components, or perform its required functions, while sharing the same hardware or software environment [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.3*] **2.** ability of two or more systems or components to exchange information **3.** capability of a functional unit to meet the requirements of a specified interface without appreciable modification [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.673

compatibility testing

- 1.** type of testing that measures the degree to which a test item can function satisfactorily alongside other independent products in a shared environment (co-existence), and where necessary, exchanges information with other systems or components (interoperability). [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.6*]

3.674

compensatory decision technique

- 1.** multiple-attribute decision technique that allows better performance in some of the attributes to compensate for lower performance in one or more of the other attributes; use of trade-offs
cf. noncompensatory decision technique, additive weighting, analytic hierarchy process, nondimensional scaling

3.675

compensatory model

- 1.** multiple-criteria decision-making model, in which a composite measure is composed of individually weighted terms and where criteria (also referred to as attribute terms) with a high value can compensate for those of a low value in proportion to each weight [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.2*]

Note 1 to entry: A compensatory model suggests that improving the more important measures (those with a higher weighting) is more likely to increase or improve the overall composite value than improving the less important ones. This model assumes that the weight (influence level) of criteria remains the same regardless of the measured level of the criteria.

3.676

competence

- 1.** demonstrated ability to apply knowledge and skills, and relevant personal attributes, as defined in the certification scheme [*ISO/IEC TR 29154:2013, Software engineering — Guide for the application of ISO/IEC 24773:2008 (Certification of software engineering professionals — Comparison framework)*, 3.3]
cf. competent

3.677

competent

- 1.** having the combination of knowledge, formal and informal skills, training, experience, and behavioral attributes required to perform a task or role

cf. competence

3.678

competent assessor

lead assessor

- 1.** assessor who has demonstrated the competencies to conduct an assessment and to monitor and verify the conformance of a process assessment [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.19], [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.2.12]

3.679

compile

- 1.** to translate a computer program expressed in a high-order language into its machine language equivalent

cf. assemble, decompile, interpret

3.680

compile-and-go

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

3.681

compiler

- 1.** computer program that translates programs expressed in a high-order language into their machine language equivalents

cf. assembler, interpreter, cross-compiler, incremental compiler, root compiler

3.682

compiler code

- 1.** computer instructions and data definitions expressed in a form that can be recognized and processed by a compiler

cf. assembly code, interpretive code, machine code

3.683

compiler directive source statement

- 1.** source statement that defines macros, or labels, or directs the compiler to insert external source statements (for example, an include statement), or directs conditional compilation, or is not described by one of the other type attributes

3.684

compiler generator

compiler compiler

metacompiler

- 1.** translator or interpreter used to construct part or all of a compiler

3.685

complaint

- 1.** record of perceived non-compliance with a service level agreement or customer dissatisfaction with service [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.2]

3.686

complete

- 1.** <documentation> including all critical information and any necessary, relevant information for the intended audience [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.2; ISO/IEC/IEEE 26511:2011 Systems and software engineering — Requirements for managers of user documentation, 4.2*]

3.687

complete ICOM code

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

3.688

complete procedure

- 1.** all those activities which commence with entry to the procedure and conclude with exit from the procedure

3.689

complete table

- 1.** decision table where for all combinations of condition entries there exists a satisfying rule [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.17*]

Note 1 to entry: In practical terms extended entry tables include limited entries and are therefore mixed entry tables. Any extended or mixed entry table can be transformed into a limited entry table.

3.690

completion code

- 1.** code communicated to a job stream processor by a batch program to influence the execution of succeeding steps in the input stream

3.691

completion criteria

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

3.692

completion time theorem

- 1.** real-time scheduling theorem

Note 1 to entry: For a set of independent periodic tasks, if each task meets its first deadline when all tasks start at the same time, the deadlines will be met for any combination of start times.

3.693

complex programmable logic device (CPLD)

- 1.** hardware component with a fully programmable AND/OR gate array

3.694

complexity

- 1.** degree to which a system's design or code is difficult to understand because of numerous components or relationships among components **2.** pertaining to any of a set of structure-based metrics that measure the attribute in (1) **3.** degree to which a system or component has a design or implementation that is difficult to understand and verify
cf. simplicity

3.695

complexity matrix

- 1.** a table used to allocate a weight to a function type [*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*]

Note 1 to entry: The matrix allocates this weight on the basis of the number of data element types in combination with the number of record types or file types referenced.

3.696

complexity of a function

- 1.** the weight allocated to a function on the basis of which a number of function points is assigned to the function
[*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.697

compliance

- 1.** doing what has been asked or ordered, as required by rule or law [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*] **2.** a general concept of conforming to a rule, standard, law, or requirement such that the assessment of compliance results in a binomial result stated as "compliant" or "noncompliant" [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

EXAMPLE: comply with a regulation

3.698

component

- 1.** entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.3.3*] **2.** one of the parts that make up a system [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*] **3.** object that encapsulates its own template, so that the template can be interrogated by interaction with the component [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.26*] **4.** specific, named collection of features that can be described by an IDL component definition or a corresponding structure in an interface repository [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*] **5.** functionally or logically distinct part of a system [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*] **c.** element, unit

Note 1 to entry: A component can be hardware or software and can be subdivided into other components. Component refers to a part of a whole, such as a component of a software product or a component of a software identification tag. The terms module, component, and unit are often used interchangeably or defined to be subelements of one another in different ways depending upon the context. The relationship of these terms is not standardized. A component can be independently managed or not, from the end-user or administrator's point of view.

3.699

component home

- 1.** meta-type that acts as a manager for instances of a specified component type [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

Note 1 to entry: Component home interfaces provide operations to manage component life cycles, and optionally, to manage associations between component instances and primary key values.

3.700

component integration test

- 1.** testing of groups of related components.

3.701

component standard

- 1.** standard that describes the characteristics of data or program components subdivided into other components

3.702

component testing

- 1.** testing of individual hardware or software components [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*;

3.703

component-aware client

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

3.704

composite key

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

Note 1 to entry: [key style]

3.705

composite measure

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

3.706

composite object

1. object expressed as a composition [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.2*]

3.707

composite task

1. task containing nested objects

3.708

composite type

1. data type each of whose members is composed of multiple data items
cf. atomic type

EXAMPLE: a data type called PAIRS whose members are ordered pairs (x,y)

3.709

composition

1. combination of two or more objects yielding a new object, at a different level of abstraction [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.1.a*] **2.** combination of two or more behaviors yielding a new behavior [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.1.b*] **3.** both the set of artifacts that constitute the unit of component implementation, and the definition of this aggregate entity [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]
cf. decomposition

3.710

computation data use

c-use

1. use of the value of a variable in any type of statement [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.4*]

3.711

computational component

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

3.712

computational container

1. container for a declared set of computational component types that has a management interface at which it can be requested to create a computational component of one of the types and add it to the container's content, delete a computational component from the container, list the computational components it currently contains, and list the computational factories it provides [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 7.1.17]

3.713

computational factory

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

3.714

computational interface template

1. interface template for either a signal interface, a stream interface, or an operation interface [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 7.1.10]

Note 1 to entry: A computational interface template comprises a signal, a stream or an operation interface signature as appropriate, a behavior specification and an environment contract specification.

3.715

computational object template

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

3.716

computational viewpoint

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

3.717

computer

1. functional unit that can perform substantial computations, including numerous arithmetic operations and logic operations without human intervention [ISO/IEC 2382:2015, *Information technology — Vocabulary*] cf. computing device

Note 1 to entry: A computer can consist of a stand-alone unit or several interconnected units.

3.718

computer center

data processing center

1. facility that includes personnel, hardware, and software, organized to provide information processing services [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.719

computer crime

1. crime committed through the use, modification, or destruction of hardware, software, or data [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.720

computer generation

1. category in a historical classification of computers based mainly on the technology used in their manufacture [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

EXAMPLE: first generation based on relays or vacuum tube, the second on transistors, the third on integrated circuits

3.721

computer graphics

1. methods and techniques for construction, manipulation, storage, and display of images by means of a computer
[*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.722

computer instruction

1. statement in a programming language, specifying an operation to be performed by a computer and the addresses or values of the associated operands **2.** loosely, any executable statement in a computer program
cf. instruction format, instruction set

EXAMPLE: Move A to B

3.723

computer language

1. language designed to enable humans to communicate with computers
cf. design language, query language, programming language

3.724

computer network

1. data processing nodes and their interconnections for the purpose of data communication [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.725

computer performance evaluation

1. engineering discipline that measures the performance of computer systems and investigates methods by which that performance can be improved

cf. system profile, throughput, utilization, workload model

3.726

computer program

1. combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions **2.** syntactic unit that conforms to the rules of a particular programming language and that is composed of declarations and statements or instructions needed for a certain function, task, or problem solution [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. software

3.727

computer program abstract

1. brief description of a computer program that provides sufficient information for potential users to determine the appropriateness of the program to their needs and resources

3.728

computer resource

1. element of a data processing system needed to perform required operations [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: storage devices, input-output units, one or more processing units, data, files, and programs

3.729

computer resource allocation

1. assignment of computer resources to current and waiting jobs
cf. dynamic resource allocation, storage allocation

EXAMPLE: the assignment of main memory, input/output devices, and auxiliary storage to jobs executing concurrently in a computer system

3.730

computer resources

- 1.** computer equipment, programs, documentation, services, facilities, supplies, and personnel available for a given purpose

cf. computer resource allocation

3.731

computer science

- 1.** branch of science and technology that is concerned with information processing by means of computers
[*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.732

computer software component (CSC)

- 1.** functionally or logically distinct part of a computer software configuration item, typically an aggregate of two or more software units

cf. computer software configuration item, software configuration item, software item

3.733

computer software configuration item (CSCI)

software configuration item (SWCI)

- 1.** aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process

cf. computer software component, hardware configuration item, configuration item, software configuration item, software item

3.734

computer system

computing system

- 1.** system containing one or more computers and associated software **2.** system containing one or more components and elements such as computers (hardware), associated software, and data [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.3*]

cf. data processing system

3.735

computer-aided (CA)

- 1.** pertaining to a technique or process in which a computer does part of the work [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.736

computer-aided design (CAD)

- 1.** use of a computer to design a device or a system, display it on a computer monitor or printer, simulate its operation, and provide statistics on its performance

Note 1 to entry: The computer is provided with data concerning the item to be designed, how it is to function, and the rules for the way in which the different components can be joined.

3.737

computer-aided software engineering (CASE)

- 1.** use of computers to aid in the software engineering process [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.2*]

cf. integrated development environment

EXAMPLE: the application of software tools to software design, requirements tracing, code production, testing, document generation, and other software engineering activities

3.738

computer-based software system (CBSS)

- 1.** software system running on a computer [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.5*]

Note 1 to entry: A CBSS can be a data processing system as seen by human users at their terminals or at equivalent machine-user-interfaces. It includes hardware and all software (system software and application software) which is necessary for realizing data processing functions required by its users

3.739

computerization

- 1.** automation by means of computers [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.740

computerize

- 1.** to automate by means of computers [*ISO/IEC 2382:2015, Information technology -- Vocabulary*]

3.741

computing center

- 1.** facility designed to provide computer services to a variety of users through the operation of computers and auxiliary hardware and through services provided by the facility's staff

3.742

computing device

- 1.** functional unit that can perform substantial computations, including numerous arithmetic operations and logic operations with or without human intervention [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.6*]

cf. computer

Note 1 to entry: A computing device can consist of a stand-alone unit, or several interconnected units. It can also be a device that provides a specific set of functions, such as a phone or a personal organizer, or more general functions such as a laptop or desktop computer.

3.743

computing system specification concepts

- 1.** visible and quantifiable abstractions of computing system characteristics having attributes in isolation and relationships in context [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.4*]

3.744

computing system tool

- 1.** computer-based tool used by a developer or maintainer organization for creating and evolving dynamic systems [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.5*]

EXAMPLE: not only traditional CASE tools, but also requirements tools, verification and validation tools, design tools, and documentation tools.

3.745

COMSEC

- 1.** communications security [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.746

concept analysis

- 1.** derivation of a system concept through the application of analysis.

cf. analysis

3.747

concept of operations

ConOps

- 1.** verbal and/or graphic statement, in broad outline, of an organization's assumptions or intent in regard to an operation or series of operations [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.12; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.11; ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.4*]

cf. operational concept

Note 1 to entry: It provides the basis for bounding the operating space, system capabilities, interfaces and operating environment. The concept of operations frequently is embodied in long-range strategic plans and annual operational plans. In the latter case, the concept of operations in the plan covers a series of connected operations to be carried out simultaneously or in succession. The concept is designed to give an overall picture of the organization operations.

3.748

concept of operations (ConOps) document

- 1.** user-oriented document that describes a system's operational characteristics from the end user's viewpoint
cf. operational concept description (OCD)

3.749

concept phase

- 1.** period of time in the system life cycle during which the user needs are identified and system concepts are described and evaluated

Note 1 to entry: precedes the requirements phase

3.750

conceptual data model

- 1.** a data model that illustrates the data groups as they are seen by the user [*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*]

3.751

conceptual model

- 1.** model of the concepts relevant to some endeavor [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.39*]

3.752

conceptual system design

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

3.753

concern

- 1.** [system] interest in a system relevant to one or more of its stakeholders [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.13; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.12; ISO/IEC/IEEE 42010:2011 Systems and software engineering — Architecture description, 3.7*] **2.** interest in something relevant to one or more of its stakeholders [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it, 3.2*]

Note 1 to entry: A concern pertains to any influence on a system in its environment, including developmental, technological, business, operational, organizational, political, economic, legal, regulatory, ecological and social influences.

3.754

conciseness

- 1.** software attributes that provide implementation of a function with a minimum amount of code

3.755

concurrency

- 1.** property of a system in which events can occur independently of each other, and hence are not ordered [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.7*]
cf. step, concurrent enabling

3.756

concurrent

- 1.** pertaining to the occurrence of two or more activities within the same interval of time, achieved either by interleaving the activities or by simultaneous execution **2.** problem, process, system, or application in which many activities happen in parallel, the order of incoming events is not usually predictable, and events often overlap
cf. parallel, simultaneous

Note 1 to entry: A concurrent system or application has many threads of control.

3.757

concurrent communication diagram

- 1.** diagram depicting a network of concurrent tasks and their interfaces in the form of asynchronous and synchronous message communication, event synchronization, and access to passive information-hiding objects

3.758

concurrent elaboration

- 1.** life cycle process instances are enacted concurrently during the project, and the information items and artefacts produced by these process instances evolve concurrently [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.1*]

Note 1 to entry: This is referred to as concurrent elaboration of information items.

3.759

concurrent enabling (of transition modes)

- 1.** multiset of transition modes is concurrently enabled if all the involved input places contain enough tokens to satisfy the sum of all of the demands imposed on them by each input arc annotation evaluated for each transition mode in the multiset [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.10*]

3.760

concurrent task architecture

- 1.** description of the concurrent tasks in a system or subsystem in terms of their interfaces and interconnections

3.761

condExpression

- 1.** syntax used in the STL both for defining simple conditions and for compounding simple conditions to define more complex conditions [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.762

condition

- 1.** measurable qualitative or quantitative attribute that is stipulated for a requirement [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.5*] **2.** description of a contingency to be considered in the representation of a problem, or a reference to other procedures to be considered as part of the condition [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.6*] **3.** true or false logical predicate [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*] **4.** logical predicate involving one or more behavior model elements [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.10*] **5.** Boolean expression containing no Boolean operators [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.6*]

3.763

condition entry

- 1.** indication of the relevance of a condition to a particular rule [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.8*]

3.764

condition stub

- 1.** list of all the conditions to be considered in the description of a problem [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.1*]

3.765

conditional information

- 1.** information supplied with every product to which it is relevant

3.766

conditional jump

1. jump that takes place only when specified conditions are met

cf. unconditional jump

3.767

Conduct Procurements

1. the process of obtaining seller responses, selecting a seller, and awarding a contract [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.768

cone of uncertainty

1. representation of how the uncertainties inherent in a project decrease over the duration of the project

3.769

confidentiality

1. degree to which a product or system ensures that data are accessible only to those authorized to have access [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.6.1]

3.770

configuration

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

3.771

configuration audit

1. in configuration management, an independent examination of the configuration status to compare with the physical configuration 2. detailed review of processes, product definition information, documented verification of compliance with requirements, and an inspection of products to confirm that products have achieved their required attributes or conform to released product configuration definition information [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.1]

3.772

configuration baseline

1. configuration information formally designated at a specific time during the life of a product, product component, service, or service component

Note 1 to entry: Configuration baselines, plus approved changes from those baselines, constitute the current configuration information.

3.773

configuration control

1. element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification
cf. change control, configuration identification, configuration status accounting

3.774

configuration control board (CCB)

1. group of people responsible for evaluating and approving or disapproving proposed changes to configuration items, and for ensuring implementation of approved changes [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] 2. qualified personnel who evaluate, for approval or disapproval, all proposed changes to the current developmental baseline [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1]

cf. configuration control board

3.775

configuration identification

1. element of configuration management, consisting of selecting the configuration items for a system and recording their functional and physical characteristics in technical documentation **2.** current approved technical documentation for a configuration item as set forth in specifications, drawings, associated lists, and documents referenced therein

cf. configuration control, configuration status accounting

3.776

configuration index

1. document used in configuration management, providing an accounting of the configuration items that make up a product

cf. configuration item development record, configuration status accounting

3.777

configuration item (CI)

1. item or aggregation of hardware, software, or both that is designated for configuration management and treated as a single entity in the configuration management process [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.14; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.13] **2.** component of an infrastructure or an item which is or will be under control of configuration management [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.7] **3.** aggregation of work products that is designated for configuration management and treated as a single entity in the configuration management process [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] **4.** entity within a configuration that satisfies an end use function and that can be uniquely identified at a given reference point [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.7] **5.** item or aggregation of software that is designed to be managed as a single entity and its underlying components, such as documentation, data structures, scripts [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.12] **6.** any system element or aggregation of system elements that satisfies an end use function and is designated by the acquirer for separate configuration control [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.1]

cf. hardware configuration item, computer software configuration item, configuration identification, critical item

Note 1 to entry: Configuration items can vary widely in complexity, size and type, ranging from an entire system including all hardware, software and documentation, to a single module or a minor hardware component.

3.778

configuration item development record

1. document used in configuration management, describing the development status of a configuration item based on the results of configuration audits and design reviews

cf. configuration index, configuration status accounting

3.779

configuration management (CM)

1. discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] **2.** technical and organizational activities, comprising configuration identification, control, status accounting and auditing [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] **3.** coordinated activities to direct and control the configuration [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities*, 3.7]

cf. baseline, change management, configuration identification, configuration control, configuration status accounting, configuration audit

3.780

configuration management authority

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

3.781

configuration management database (CMDB)

1. specific type of repository for CM information, usually a data store, used to record attributes of configuration items, and the relationships between configuration items, throughout their lifecycle [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] **2.** database containing all the relevant details of each configuration item and details of the important relationships between them [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.8]
cf. configuration management system

3.782

configuration management system

1. discipline of identifying the components of a continually evolving system to control changes to those components and maintaining integrity and traceability throughout the life cycle **2.** a subsystem of the overall project management system. It is a collection of formal documented procedures used to apply technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a product, result, service, or component; control any changes to such characteristics; record and report each change and its implementation status; and support the audit of the products, results, or components to verify conformance to requirements. It includes the documentation, tracking systems, and defined approval levels necessary for authorizing and controlling changes [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. configuration management database (CMDB)

3.783

configuration status accounting (CSA)

1. element of configuration management that consists of the recording and reporting of information needed to manage a configuration effectively [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities*, 3.10]

cf. configuration control, configuration identification, configuration index, configuration item, development record.

Note 1 to entry: This information includes a listing of the approved configuration identification, the status of proposed changes to the configuration, and the implementation status of approved changes.

3.784

conflict

1. change in one version of a file that cannot be reconciled with the version of the file to which it is applied

Note 1 to entry: can occur when versions from different branches are merged or when two committers work concurrently on the same file.

3.785

conflict management

1. handling, controlling, and guiding a conflictual situation to achieve a resolution [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.786

conformance

1. within the quality management system, a general concept of delivering results that fall within the limits that define acceptable variation for a quality requirement [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: For conformance to be meaningful, the specified requirements accurately represent stakeholder requirements.

3.787

conformance point

1. reference point at which behavior can be observed for the purposes of conformance testing [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 10.7]

3.788

conformance work

1. In the cost of quality framework, conformance work is done to compensate for imperfections that prevent organizations from completing planned activities correctly as essential first-time work. Conformance work consists of actions that are related to prevention and inspection [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.789

conformity assessment

1. demonstration that specified requirements relating to a product, process, system, person or body are fulfilled [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.21]

3.790

conformity evaluation

1. systematic examination of the extent to which a product, process, or device fulfills specified requirements [*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*, 4.1.4]

3.791

conformity evaluation report

1. document that describes the conduct and results of the evaluation carried out for a Ready to Use software product (RUSP) [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.4]

3.792

ConnectionPath

1. pathway for the propagation of information [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.793

connectivity

1. capability of a system or device to be attached to other systems or devices without modification [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.794

ConOps

1. concept of operations [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

3.795

consecutive

1. pertaining to the occurrence of two sequential events or items without the intervention of any other event or item; that is, one immediately after the other

3.796

consequence

1. outcome of an event [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management*, 3.1] 2. effect (change or non-change), usually associated with an event or condition or with the system and usually allowed, facilitated, caused, prevented, changed, or contributed to by the event, condition, or system [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.4.1] 3. outcome of an event affecting objectives [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.3]

Note 1 to entry: The outcome can be a loss or a benefit, or neither, and can be expressed qualitatively or quantitatively.

3.797

consistency

1. degree of uniformity, standardization, and freedom from contradiction among the documents or parts of a system or component **2.** software attributes that provide uniform design and implementation techniques and notations

cf. traceability

3.798

consistent

1. without internal conflicts [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.3]

3.799

consistent state

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

EXAMPLE: The Functional User Requirement is to print a check and mark the appropriate account as paid. If only part of the Functional User Requirement is completed (e.g., only printing the check or only marking it as paid), the application is not in a consistent state. The printing of a check without marking the account as paid causes an inconsistency in the application as does marking it as paid without printing it.

3.800

consolidation of an entitlement

1. process of combining two or more entitlements into a single, unified entitlement [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.5]

Note 1 to entry: Entitlements can be consolidated to simplify understanding of the current position or as the result of a licensor negotiation. The entitlement schema enables the recording of entitlement consolidations.

3.801

consolidation tag

1. type of tag used to represent a grouping of multiple other tags

EXAMPLE: In entitlement management, a consolidation tag can be used to facilitate subsequent creation of children tags with entitlement quantities which do not match the original granted entitlements

3.802

constant

1. quantity or data item whose value cannot change **2.** instance whose identity is known at the time of writing [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.40] **3.** specification that an attribute or participant property value, once assigned, shall not be changed, or that an operation shall always provide the same output argument values given the same input argument values. [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.40] **4.** numeric or string value that does not change during program execution **5.** reference name for a numerical, lexical, or externally specified fixed (unchanging) value [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*] *cf.* variable figurative constant, literal

EXAMPLE: the data item FIVE, with an unchanging value of 5

Note 1 to entry: The identity of a constant state class instance is represented by #K, where K is an integer or a name.

3.803

constant dollar analysis

1. addressing inflation and deflation by using cash flow amounts that represent money values which are referenced to a fixed time (typically, the beginning of a project)

cf. actual dollar analysis

3.804

constant-failure period

1. period of time in the life cycle of a system or component during which hardware failures occur at an approximately uniform rate

cf. early-failure period, wearout-failure period, bathtub curve

3.805

constituent configuration item

1. individual item to be controlled that is a constituent (part) of a larger configuration item, such as a reference model, hardware prototype or software build [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]

3.806

constraint

1. limitation or implied requirement that constrains the design solution or implementation of the systems engineering process and is not changeable by the enterprise [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*] **2.** restriction on software life cycle process (SLCP) development **3.** rule that specifies a valid condition of data [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.41*] **4.** responsibility that is a statement of facts that are required to be true in order for the constraint to be met [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.41*] **5.** restriction on the value of an attribute or the existence of any object based on the value or existence of one or more others [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*] **6.** externally imposed limitation on system requirements, design, or implementation or on the process used to develop or modify a system [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.6*] **7.** a limiting factor that affects the execution of a project, program, portfolio, or process [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: A constraint is a factor that is imposed on the solution by force or compulsion and can limit or modify the design changes.

3.807

construct

1. concept, such as the abstract idea, image, underlying theme, or subject matter, that one wishes to measure using process assessments [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.5*]

Note 1 to entry: In process measurement frameworks, constructs (also refers to latent constructs) are theoretical concepts, such as the process quality characteristics and process attributes. The meaning that one assigns to a construct is its theoretical definition, which describes its distinct dimensions (facets).

3.808

construction

1. process of writing, assembling, or generating assets [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*] **2.** activity in software development consisting of detailed design, coding, unit testing, and debugging

Note 1 to entry: the collection of activities focused on creating source code

3.809

consumer

1. organization or person who buys the software package **2.** event sink [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

cf. event sink

3.810

consumer object

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

3.811

consumer software package

1. COTS software product designed and sold for end users to carry out identified functions; the software and its associated documentation are packaged for sale as a unit

3.812

container

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

Note 1 to entry: Containers provide the run-time execution environment for CORBA component implementations.

3.813

container interface

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

3.814

content

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

3.815

content consistency

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

3.816

content coupling

1. type of coupling in which some or all of the contents of one software module are included in the contents of another module

cf. common-environment coupling, control coupling, data coupling, hybrid coupling, pathological coupling

3.817

content management

1. control of units of information with their metadata, to allow selective reuse in documents or information items with variable structures and formats [ISO/IEC/IEEE 26511:2011 *Systems and software engineering — Requirements for managers of user documentation*, 4.4]

Note 1 to entry: Content management for user documentation means management of help topics, explanations of concepts, troubleshooting procedures, compliance statements, and variables such as the names and host platforms of software products, with metadata tags that are applied to format output.

3.818

context

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

3.819

context completeness

1. degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in all the specified contexts of use [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.5.1]

EXAMPLE: The extent to which software is usable using a small screen, with low network bandwidth, by a non-expert user, and in a fault-tolerant mode (e.g. no network connectivity).

Note 1 to entry: Context completeness can be specified or measured either as the degree to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, freedom from risk and satisfaction in all the intended contexts of use, or by the presence of product properties that support use in all the intended contexts of use.

3.820

context coverage

1. degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in both specified contexts of use and in contexts beyond those initially explicitly identified [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.5]

Note 1 to entry: Context of use is relevant to both quality in use and some product quality (sub)characteristics (where it is referred to as specified conditions).

3.821

context diagram

1. diagram that presents the context of the top-level function of an IDEF0 model, whose diagram number is a-n, where 0#n#9 [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.31]

Note 1 to entry: The one-box A-0 context diagram is a required context diagram; those with diagram numbers A-1, A-2, ..., A-9 are optional context diagrams.

3.822

context diagrams

1. a visual depiction of the product scope showing a business system (process, equipment, computer system, etc.), and how people and other systems (actors) interact with it [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.823

context of use

1. users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a system, product, or service is used [*ISO/IEC 25063:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: Context of use description*] **2** users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.2]

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

[SOURCE: ISO 9241-11:1998].

3.824

context-sensitive help

1. type of on-screen documentation in which the information that is displayed depends upon the user's view of the software [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.10]

cf. embedded documentation, printed documentation

3.825

contextual schema

1. formal description of the boundary of the context of use where data models are applied [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.4]

Note 1 to entry: It is a high-level description of the business informational needs. It is more general than a conceptual model, as it includes a holistic vision of a (system) context of the architecture.

3.826

contiguous allocation

- 1.** storage allocation technique in which programs or data to be stored are allocated a block of storage of equal or greater size, so that logically contiguous programs and data are assigned physically contiguous storage locations
cf. paging (1)

3.827

contingency

- 1.** an event or occurrence that could affect the execution of the project that may be accounted for with a reserve [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.828

contingency plan

- 1.** plan for dealing with a risk factor, if it becomes a problem

3.829

contingency reserve

- 1.** budget within the cost baseline or performance measurement baseline that is allocated for identified risks that are accepted and for which contingent or mitigating responses are developed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.830

contingent response strategies

- 1.** responses provided which may be used in the event that a specific trigger occurs [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.831

continual improvement

- 1.** recurring activity to increase the ability to fulfill service requirements

3.832

continual process improvement

- 1.** ongoing cycle of process improvement programs to strengthen and improve the processes supporting business and include one or several improvement projects or initiatives, which can be implemented in series or in parallel [*ISO/IEC TR 33014:2013 Information technology — Process assessment — Guide for process improvement, 3.1*]

3.833

continuing professional development (CPD)

- 1.** set of activities undertaken by an individual professional to maintain professional competence [*ISO/IEC TR 29154:2013, Software engineering — Guide for the application of ISO/IEC 24773:2008 (Certification of software engineering professionals — Comparison framework), 3.4*]

3.834

continuous forms

- 1.** forms produced in continuous lengths during the manufacturing process and intended primarily for use with sprocket-hole transporting mechanisms [*ISO 3535:1977 Forms design sheet and layout chart, 4.1*]

3.835

continuous iteration

- 1.** loop that has no exit

3.836

continuous representation

- 1.** capability maturity model structure wherein capability levels provide a recommended order for approaching process improvement within each specified process area

3.837

continuous risk management

- 1.** process of analyzing the progress of a planned activity, project, or program on a periodic, ongoing basis and handling identified risk factors

Note 1 to entry: includes developing options and fallback positions to permit alternative solutions to reduce the impact if a risk factor becomes a problem.

3.838

contract

1. binding agreement between two parties, especially enforceable by law, or a similar internal agreement wholly within an organization [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.8*]
2. agreement governing part of the collective behavior of a set of objects [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.1*]
3. a mutually binding agreement that obligates the seller to provide the specified product or service or result and obligates the buyer to pay for it [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: A contract is an agreement between two or more parties regarding a course of action. The formality of a contract can range from a simple informal oral description to a formal written instrument.

3.839

contract administration

1. process of managing the contract and the relationship between the acquirer and supplier, including reviewing and documenting how the supplier is performing or has performed; establishing required corrective actions; and managing contract changes

3.840

contract change control system

1. the system used to collect, track, adjudicate, and communicate changes to a contract [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.841

contract management plan

1. document that describes how a specific agreement will be administered to monitor delivery of required documentation and performance of the statement of work, to evaluate performance, and to control changes

3.842

contract work breakdown structure (CWBS)

1. portion of the overall work breakdown structure applicable to a contract, developed and maintained by the supplier

3.843

contractual context

1. knowledge that a particular contract is in place, and thus that a particular behavior of a set of objects is required [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.2.3*]

3.844

contractual requirement

1. result of the analysis and refinement of customer requirements into a set of requirements suitable to be included in one or more solicitation packages, formal contracts, or supplier agreements between the acquirer and other appropriate organizations
cf. acquirer, customer requirement

3.845

contravariance

1. rule governing the overriding of a property and requiring that the set of values acceptable for an input argument in the overriding property shall be a superset (includes the same set) of the set of values acceptable for that input argument in the overridden property, and the set of values acceptable for an output argument in the overriding property shall be a subset (includes the same set) of the set of values acceptable for that output argument in the overridden property [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.42*]

3.846

control

1. in engineering, the monitoring of system output to compare with expected output and taking corrective action when the actual output does not match the expected output 2. comparing actual performance with planned performance, analyzing variances, assessing trends to effect process improvements, evaluating possible alternatives, and recommending appropriate corrective action as needed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 3. in an IDEF0 model, a condition or set of conditions required for a function to produce correct output [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 2.1.32*]

3.847

control account

1. a management control point where scope, budget, actual cost, and schedule are integrated and compared to earned value for performance measurement [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] cf. work package

3.848

control arrow

1. arrow or arrow segment that expresses IDEF0 control [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.33*]

Note 1 to entry: That is, an object type set whose instances establish a condition or set of conditions required for a function to produce correct output. The arrowhead of a control arrow is attached to the top side of a box.

3.849

control chart

1. a graphic display of process data over time and against established control limits, which has a centerline that assists in detecting a trend of plotted values toward either control limit [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.850

control clustering

1. task-structuring criterion by which a control object is combined into a task with the objects it controls

3.851

control communications

1. the process of monitoring and controlling communications throughout the entire project life cycle to ensure the information needs of the project stakeholders are met [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.852

Control Costs

1. the process of monitoring the status of the project to update the project costs and managing changes to the cost baseline [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.853

control coupling

1. type of coupling in which one software module communicates information to another module for the explicit purpose of influencing the latter module's execution

cf. common-environment coupling, content coupling, data coupling, hybrid coupling, pathological coupling

3.854

control data

1. data that select an operating mode, direct the sequential flow of a program, or otherwise directly influence the operation of software

EXAMPLE: a loop control variable

3.855

control field

- 1.** field comprising one or more input variables whose change in value, or lack of change, between successive logical records affects the flow of control through the main procedure

3.856

control flow

flow of control

- 1.** sequence in which operations are performed during the execution of a computer program **2.** sequence in which operations are performed during the execution of a test item [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.7*]

cf. data flow

3.857

control flow diagram

- 1.** diagram that depicts the set of all possible sequences in which operations can be performed during the execution of a system or program

cf. data flow diagram, call graph, structure chart

Note 1 to entry: Types include box diagram, flowchart, input-process-output chart, state diagram.

3.858

control flow sub-path

- 1.** sequence of executable statements within a test item [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.8*]

3.859

control Information

- 1.** data that turns on or off one or more processes of an application or that influences the operation of a transaction [*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.** data that influences an elementary process by specifying what, when or how data is to be processed [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.11*]

3.860

control limits

- 1.** the area composed of three standard deviations on either side of the centerline, or mean, of a normal distribution of data plotted on a control chart, which reflects the expected variation in the data [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. specification limits

3.861

control loopback

feedback

- 1.** loopback of output from one function to be control for another function in the same diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.34*]

3.862

control procurements

- 1.** the process of managing procurement relationships, monitoring contract performance, and making changes and corrections as appropriate [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.863

control quality

- 1.** The process of monitoring and recording results of executing the quality activities to assess performance and recommend necessary changes [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.864

control risks

monitor and control risks

- 1.** the process of implementing risk response plans, tracking identified risks, monitoring residual risks, identifying new risks, and evaluating risk process effectiveness throughout the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.865

Control Schedule

- 1.** the process of monitoring the status of project activities to update project progress and manage changes to the schedule baseline to achieve the plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.866

Control Scope

- 1.** the process of monitoring the status of the project and product scope and managing changes to the scope baseline [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.867

control stakeholder engagement

- 1.** the process of monitoring overall project stakeholder relationships and adjusting strategies and plans for engaging stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.868

control statement

- 1.** program statement that selects among alternative sets of program statements or affects the order in which operations are performed

cf. assignment statement, declaration

EXAMPLE: if-then-else, case

3.869

control store

- 1.** in a microprogrammed computer, the computer memory in which microprograms reside

cf. microword, nanostore

3.870

control task

- 1.** task that makes decisions to control other tasks' execution

3.871

controller

- 1.** device or computer chip that interfaces with a peripheral device

3.872

controller area network (CAN)

- 1.** high-integrity bus system for networking intelligent devices within a system

Note 1 to entry: commonly used in embedded networks for vehicles or medical equipment

3.873

convention

- 1.** requirement employed to prescribe a disciplined, uniform approach to providing consistency in a software product, that is, a uniform pattern or form for arranging data

cf. practice, standard

3.874

conversational

- 1.** pertaining to an interactive system or mode of operation in which the interaction between the user and the system resembles a human dialog

cf. batch, interactive, online, real time

3.875

conversion

- 1.** modification of existing software to enable it to operate with similar functional capability in a different environment

EXAMPLE: converting a program from FORTRAN to Ada, converting a program that runs on one computer to run on another

3.876

conversion functionality

- 1.** transactional or data functions provided to convert data or provide other user-specified conversion requirements [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.12*]

Note 1 to entry: Conversion functionality exists only during the development or enhancement of an application.

3.877

convertibility

- 1.** the ability to convert the results from applying two or more FSM methods in the measurement of a functional size of the same set of functional user requirements [*ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods, 3.3*]

3.878

cookie

- 1.** small file that is stored in and retrieved from user web storage to maintain state information, including identification of users and transaction coherency [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.7*]

Note 1 to entry: Web sites store/retrieve cookies from user client systems to maintain state information including identification of users and transaction coherency.

3.879

coordinated interactions

- 1.** pair of interactions in which the first interaction is generated by a unit as a stimulus to some other unit, in expectation that a second interaction will be generated as a response from that other unit (or even a third unit) to benefit the initiating unit for a particular purpose [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.11*]

Note 1 to entry: One unit controls the initiation of this exchange, subordinating the other unit(s) to itself. Such a subordinating exchange is possible even if the subordinated unit generated the stimulus that triggered the initiating unit's behavior.

3.880

copy

- 1.** to read data from a source, leaving the source data unchanged, and to write the same data elsewhere in a physical form that can differ from that of the source **2.** result of a copy process as in (1)
cf. move

EXAMPLE: to copy data from a magnetic disk onto a magnetic tape

3.881

copyright

- 1.** exclusive right granted to the owner of an original work of authorship, which is fixed in any tangible medium of expression, to reproduce, publish, perform, and/or sell the work

3.882

COQ

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

3.883

CORBA

- 1.** Common Object Request Broker Architecture [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function, 4; ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 5*]

3.884

core

- 1.** processing unit in a computer or processor which manages instructions, data, and operations

3.885

core report

- 1.** document for providing descriptions of the process and outcomes of the benchmarking activity [*ISO/IEC 29155-3:2015 Systems and software engineering — Information technology project performance benchmarking framework — Part 3: Guidance for reporting*]

Note 1 to entry: Two kinds of core reports (i.e. executive summary and detailed report) are often produced for reporting results of an instance of benchmarking activity.

3.886

routine

- 1.** routine that begins execution at the point at which operation was last suspended, and that is not required to return control to the program or subprogram that called it
cf. subroutine

3.887

corporate board or equivalent body

- 1.** person or group of people who assumes legal responsibility for conducting or controlling an organization at the highest level [*ISO/IEC 19770-1:2012 Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance, 3. 3*]

3.888

corporate governance of IT

- 1.** at the level of top management, establishment of strategy and policy for the use of IT, and organizational control of the use of IT

3.889

correctability

- 1.** degree of effort required to correct software defects and to cope with user complaints

3.890

corrective action

- 1.** an intentional activity that realigns the performance of the project work with the project management plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** action to eliminate the cause or reduce the likelihood of recurrence of a detected nonconformity or other undesirable situation

3.891

corrective maintenance

- 1.** the reactive modification of a software product performed after delivery to correct discovered problems [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.2; ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.13*] **2.** maintenance performed to correct faults in hardware or software

Note 1 to entry: The modification repairs the software product to satisfy requirements.

3.892

correctness

- 1.** degree to which a system or component is free from faults in its specification, design, and implementation **2.** degree to which software, documentation, or other items meet specified requirements **3.** degree to which software, documentation, or other items meet user needs and expectations, whether specified or not

3.893

COS

1. common object services [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.894

COSMIC

1. Common Software Measurement International Consortium [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method*]

3.895

cost aggregation

1. summing the lower-level cost estimates associated with the various work packages for a given level within the project's WBS or for a given cost control account [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.896

cost avoidance

1. revenue (positive cash flow) that results from decreasing expenses, rather than from increasing income

3.897

cost baseline

1. the approved version of the time-phased project budget, excluding any management reserves, which can be changed only through formal change control procedures and is used as a basis for comparison to actual results [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.898

cost basis

acquisition cost

1. entire cost to acquire an asset

Note 1 to entry: includes the purchase price, delivery, installation, and any other costs to put the asset into service

3.899

cost constraint

1. limitation or restraint placed on the project budget, such as funds available over time

3.900

cost function

1. objective function that characterizes the cost associated with different values of the decision variable c_f income function

3.901

cost management plan

1. a component of a project or program management plan that describes how costs will be planned, structured, and controlled [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.902

cost of quality (COQ)

1. a method of determining the costs incurred to ensure quality. Prevention and appraisal costs (cost of conformance) include costs for quality planning, quality control (QC), and quality assurance to ensure compliance to requirements (i.e., training, QC systems, etc.). Failure costs (cost of non-conformance) include costs to rework products, components, or processes that are non-compliant, costs of warranty work and waste, and loss of reputation [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.903

cost performance baseline

1. time-phased budget under change control, used to compare actual expenditures to planned expenditures

Note 1 to entry: used to determine if preventive or corrective action is needed to meet the project objectives

3.904

cost performance index (CPI)

1. a measure of the cost efficiency of budgeted resources expressed as the ratio of earned value to actual cost [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.905

cost plus award fee contracts

1. a category of contract that involves payments to the seller for all legitimate actual costs incurred for completed work, plus an award fee representing seller profit [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.906

cost plus award fee contracts (CPAF)

1. a category of contract that involves payments to the seller for all legitimate actual costs incurred for completed work, plus an award fee representing seller profit [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.907

cost plus fixed fee contract (CPFF)

1. a type of cost-reimbursable contract where the buyer reimburses the seller for the seller's allowable costs (allowable costs are defined by the contract) plus a fixed amount of profit (fee) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.908

cost plus incentive fee contract (CPIF)

1. a type of cost-reimbursable contract where the buyer reimburses the seller for the seller's allowable costs (allowable costs are defined by the contract), and the seller earns its profit if it meets defined performance criteria [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.909

cost variance (CV)

1. the amount of budget deficit or surplus at a given point in time, expressed as the difference between the earned value and the actual cost [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.910

cost-benefit analysis

1. a financial analysis tool used to determine the benefits provided by a project against its costs [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.911

cost-plus-fee (CPF)

1. contract in which the acquirer reimburses the supplier's allowable costs for performing the contract work and also pays a fee

3.912

cost-plus-fixed-fee (CPFF) contract

1. a type of cost-reimbursable contract where the buyer reimburses the seller for the seller's allowable costs (allowable costs are defined by the contract) plus a fixed amount of profit (fee) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.913

cost-plus-incentive-fee (CPIF) contract

1. a type of cost-reimbursable contract where the buyer reimburses the seller for the seller's allowable costs (allowable costs are defined by the contract), and the seller earns its profit if it meets defined performance criteria [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.914

cost-reimbursable contract

1. a type of contract involving payment (reimbursement) by the buyer to the seller for the seller's actual costs, plus a fee typically representing seller's profit. Cost-reimbursable contracts often include incentive clauses where, if the seller meets or exceeds selected project objectives, such as schedule targets or total cost, then the seller

receives from the buyer an incentive or bonus payment [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.915

COTS

1. Commercial-Off-The-Shelf [*ISO/IEC 90003:2014 Software engineering — Guidelines for the application of ISO 9001:2008 to computer software, 3.4*]

3.916

counter

1. variable used to record the number of occurrences of a given event during the execution of a computer program

EXAMPLE: a variable that records the number of times a loop is executed

3.917

counting rule

1. conditions and procedures under which the measurement value is obtained

3.918

counting scope

1. set of Functional User Requirements to be included in the function point count [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.14*]

3.919

coupling

1. manner and degree of interdependence between software modules **2.** strength of the relationships between modules. **3.** measure of how closely connected two routines or modules are **4.** in software design, a measure of the interdependence among modules in a computer program [*ISO/IEC TR 19759:2016 Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK) 2.1.4*,
cf. cohesion]

Note 1 to entry: Types include common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling, and pathological coupling.

3.920

CPAF

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

3.921

CPC

1. computer program component
cf. computer software component

3.922

CPCI

1. computer program configuration item
cf. computer software configuration

3.923

CPD

1. continuing professional development [*ISO/IEC TR 29154:2013, Software engineering — Guide for the application of ISO/IEC 24773:2008 (Certification of software engineering professionals — Comparison framework), 3.4*] **2.** capability production document [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.924

CPF

1. cost-plus-fee

3.925

CPFF

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

3.926

CPG

1. clock pulse generator

3.927

CPI

1. cost performance index [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** critical program information [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.928

CPIF

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

3.929

CPLD

1. complex programmable logic device

3.930

CPM

1. critical path method **2.** Counting Practices International Standard [*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*]

3.931

CPPC

1. cost plus percentage of cost

3.932

CPU

1. central processing unit [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*]

3.933

crash

1. sudden and complete failure of a computer system or component

cf. hard failure

3.934

crashing

1. a technique used to shorten the schedule duration for the least incremental cost by adding resources [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. fast tracking, schedule compression

3.935

create WBS (work breakdown structure)

1. the process of subdividing project deliverables and project work into smaller, more manageable components [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.936

creation

1. of an <X>, instantiating by an action of objects in the model [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.18*]

cf. introduction

3.937

crisis

1. critical state of affairs in which a decisive, probably undesirable outcome is impending

3.938

crisis management

- 1.** steps to take when a contingency plan does not solve the associated problem

3.939

criteria

- 1.** standards, rules, or tests on which a judgment or decision can be based, or by which a product, service, result, or process can be evaluated [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** specific data items identified as contents of information items for appraising a factor in an evaluation, audit, test or review [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.3]

3.940

critical chain method

- 1.** a schedule method that allows the project team to place buffers on any project schedule path to account for limited resources and project uncertainties [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.941

critical design review (CDR)

- 1.** review conducted to verify that the detailed design of one or more configuration items satisfy specified requirements; to establish the compatibility among the configuration items and other items of equipment, facilities, software, and personnel; to assess risk areas for each configuration item; and, as applicable, to assess the results of producibility analyses, review preliminary hardware product specifications, evaluate preliminary test planning, and evaluate the adequacy of preliminary operation and support documents **2.** review as in (1) of any hardware or software component

3.942

critical information

- 1.** information describing the safe use of the software, the security of the information created with the software, or the protection of the sensitive personal information created by or stored with the software [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.11; *ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.6]

3.943

critical item

- 1.** in configuration management, an item within a configuration item that, because of special engineering or logistic considerations, requires an approved specification to establish technical or inventory control at the component level

3.944

critical path

- 1.** the sequence of activities that represents the longest path through a project, which determines the shortest possible duration [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] *cf.* critical path methodology

3.945

critical path activity

- 1.** any activity on the critical path in a project schedule [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.946

critical path method (CPM)

- 1.** a method used to estimate the minimum project duration and determine the amount of scheduling flexibility on the logical network paths within the schedule model [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.947

critical piece first

- 1.** system development approach in which the most critical aspects of a system are implemented first
cf. bottom-up, top-down

Note 1 to entry: The critical piece can be defined in terms of services provided, degree of risk, difficulty, or other criteria.

3.948

critical range

- 1.** metric values used to classify software into the categories of acceptable, marginal, or unacceptable [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.2*]

3.949

critical section

- 1.** section of a task's internal logic that is executed mutually exclusively with other tasks

3.950

critical system

- 1.** system having the potential for serious impact on the users or environment, due to factors including safety, performance, and security [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.22*] **2.** those items (e.g. functions, parts, software, characteristics, processes) having significant effect on the product realization and use of the product — including safety, performance, form, fit, function, producibility, service life, etc. — that require specific actions to ensure they are adequately managed [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 3.2*]

EXAMPLE: safety critical items, fracture critical items, mission critical items, key characteristics

3.951

critical value

- 1.** metric value of a validated metric that is used to identify software that has unacceptable quality [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.3*]

3.952

criticality

- 1.** degree of impact that a requirement, module, error, fault, failure, or other item has on the development or operation of a system [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*].

3.953

cross-assembler

- 1.** assembler that executes on one computer but generates machine code for a different computer

3.954

cross-compiler

- 1.** compiler that executes on one computer but generates machine code for a different computer

3.955

cross-reference generator

cross-referencer

- 1.** software tool that accepts as input the source code of a computer program and produces as output a listing that identifies each of the program's variables, labels, and other identifiers and indicates which statements in the program define, set, or use each one

3.956

cross-reference list

- 1.** list that identifies each of the variables, labels, and other identifiers in a computer program and indicates which statements in the program define, set, or use each one

3.957

cross-reference tool

- 1.** software maintenance tool that lets the user determine where a variable is used or where a particular procedure is called on

3.958

CRUD

- 1.** create, read, update, delete [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.959

CSA

- 1.** configuration status accounting [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities*, 3.10]

3.960

CSC

- 1.** computer software component

3.961

CSCI

SWCI

- 1.** computer software configuration item

3.962

CSF

- 1.** critical success factor [*ISO/IEC TR 14471:2007 Information technology — Software engineering — Guidelines for the adoption of CASE tools*, 2.2]

3.963

CSIV2

- 1.** common secure interoperability, version 2 [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 5]

3.964

CSS

- 1.** cascading style sheets [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.2.1; *ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.965

CSS2

- 1.** cascading stylesheets level 2 [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.2.2]

3.966

CT

- 1.** communication technology

3.967

cumulative flow diagram (CFD)

- 1.** a chart indicating features completed over time, plus features in development, and those features in the backlog [*Software Extension to the PMBOK® Guide Fifth Edition*]

Note 1 to entry: may indicate features at some intermediate milestones, such as features designed but not yet constructed

3.968

curriculum standard

- 1.** standard that describes the characteristics of a course of study on a body of knowledge that is offered by an educational institution

3.969

custom software

1. software product developed for a specific application from a user requirements specification [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.3]

3.970

customer

beneficiary
purchaser

1. organization or person that receives a product or service [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.9; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.16; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.14] 2. the person or organization that will use the project's product or service or result [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 3. organization or part of an organization that receives a service or services or products of the application management organization [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.14] 4. person or organization that could or does receive a product or a service that is intended for or required by this person or organization [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.22]
cf. acquirer, buyer, stakeholder, user

EXAMPLE: consumer, client, end-user, retailer, receiver or product or service from an internal process, an organization within the same company as the developing organization (e.g., System Management), a company or entity external to the developing company, a higher -level project, or some combination of these

Note 1 to entry: A customer can be internal or external to the organization. This is the entity to whom the system developer must provide proof that the system developed satisfies the system requirements specified. Customers are a subset of stakeholders. An application management organization can have customers that are internal or external business information management organizations and other application management organizations. A user or end user is a person that actually uses the application software, where a customer is a person or organization that decides about and acquires the products or services. The customer or user organization is, in its relationships with application management, represented by business information management.

3.971

customer requirement

1. result of eliciting, consolidating, and resolving conflicts among the needs, expectations, constraints, and interfaces of the product's relevant stakeholders in a way that is acceptable to the customer

3.972

customer satisfaction

1. within the quality management system, a state of fulfillment in which the needs of a customer are met or exceeded for the customer's expected experiences as assessed by the customer at the moment of evaluation [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.973

customization

1. adaptation of a software or documentation product to the needs of a particular audience [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.12]

3.974

cut-off date

1. date after which changes to the software are reflected in the next, rather than the current, software release or issue of the documentation

3.975

cutover

1. transfer of functions of a system to its successor at a given moment. *ISO/IEC 2382:2015, Information technology — Vocabulary*

3.976

CV

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

3.977

CVE

1. Common Vulnerabilities and Exposures [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.978

CVSS

1. Common Vulnerability Scoring System [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.979

CWBS

1. contract work breakdown structure [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.980

cycle

1. period of time during which a set of events is completed **2.** set of operations that is repeated regularly in the same sequence, possibly with variations in each repetition

cf. software development cycle, software life cycle

3.981

cycle stealing

1. process of suspending the operation of a central processing unit for one or more cycles to permit the occurrence of other operations, such as transferring data from main memory in response to an output request from an input/output controller

3.982

cyclic search

1. storage allocation technique in which each search for a suitable block of storage begins with the block following the one last allocated

3.983

dangerous condition

1. state of a system which, in combination with some states of the environment, will result in adverse consequence [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.4]

Note 1 to entry: A hazardous situation in ISO/IEC Guide 51 and IEC 61508-4 is an instance of a dangerous condition. A concept of dangerous conditions is introduced in order to cover not only hazardous situations in the safety context but also errors in the reliability, integrity, confidentiality, or dependability contexts and other states of a system which can lead to adverse consequences. Occurrences of failures in the context of reliability often, but not always, lead to dangerous conditions. A dangerous condition therefore has attributes, at least, a) the associated adverse consequences, b) the trigger events that lead to the dangerous condition, and c) the trigger events that lead to the adverse consequences from the dangerous condition.

3.984

dark matter

1. the work missed in the original project plan that is required to complete the deliverable product [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.985

data

1. representation of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automatic means **2.** collection of values assigned to base measures, derived measures and/or indicators [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.4] **3.** representations of information dealt with by information systems and users thereof [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 3.2.1] **4.** reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or processing

[ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.4]

cf. data type

3.986

data abstraction

1. process of extracting the essential characteristics of data by defining data types and their associated functional characteristics and disregarding representation details 2. result of the process in (1)

cf. encapsulation, information hiding

3.987

data analysis

1. systematic investigation of the data and their flow in a real or planned system [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.988

data attribute

1. smallest parcel of information, within an identified data group, carrying a meaning from the perspective of the software's functional user requirements [ISO/IEC 19761:2011 *Software engineering — COSMIC: a functional size measurement method*, 3.4]

3.989

data bank

1. set of data related to a given subject and organized in such a way that it can be consulted by subscribers [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.990

data breakpoint

storage breakpoint

1. breakpoint that is initiated when a specified data item is accessed

cf. code breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint

3.991

data buffer register (DBR)

1. region of a physical memory storage used to temporarily store data while it is being moved, e.g., from input to processing

3.992

data characteristic

1. inherent, possibly accidental, trait, quality, or property of data.

EXAMPLE: arrival rates, formats, value ranges, or relationships between field values

3.993

data communication

1. transfer of data among functional units according to sets of rules governing data transmission and the coordination of the exchange [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.994

data coupling

input-output coupling

1. type of coupling in which output from one software module serves as input to another module

cf. common-environment coupling, content coupling, control coupling, hybrid coupling, pathological coupling

3.995

data date

as-of date

time-now date

1. a point in time when the status of the project is recorded [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.996

data declaration source statement

1. source statement that reserves or initializes memory at compilation time

3.997

data definition

variable definition

1. statement where a variable is assigned a value [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.9*]

3.998

data definition c-use pair

1. data definition and subsequent computation data use, where the data use uses the value defined in the data definition [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.10*]

3.999

data definition p-use pair

1. data definition and subsequent predicate data use, where the data use uses the value defined in the data definition [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.11*]

3.1000

data definition-use pair

1. data definition and subsequent data use, where the data use uses the value defined in the data definition [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.12*]

3.1001

data dictionary

1. collection of information about data such as name, description, creator, owner, provenance, translation in different languages, and usage [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.7*]

3.1002

data element

data item

1. unique, user-recognizable, non-repeated field in a BFC [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, 3.3*]

Note 1 to entry: A data element can be a character string, or a digital or graphical element in a BFC. When 'data elements' are indicated for a BFC, the number of data elements is always greater than 0.

3.1003

data element type (DET)

1. unique, user-recognizable, non-repeated attribute [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.15*] **2.** the most elementary form of data as seen by the user that serves for controlling, recording, or transferring information [*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.** a unique, user-recognizable, non-recursive item of information [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

3.1004

data exception

1. exception that occurs when a program attempts to use or access data incorrectly

cf. addressing exception, operation exception, overflow exception, protection exception, underflow exception

3.1005

data file

1. set of related data records treated as a unit [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.7*]

3.1006

data flow

1. sequence in which data transfer, use, and transformation are performed during the execution of a computer program
cf. control flow

3.1007

data flow diagram (DFD)

data flowchart

data flow graph

1. diagram that depicts data sources, data sinks, data storage, and processes performed on data as nodes, and logical flow of data as links between the nodes
cf. control flow diagram, data structure diagram

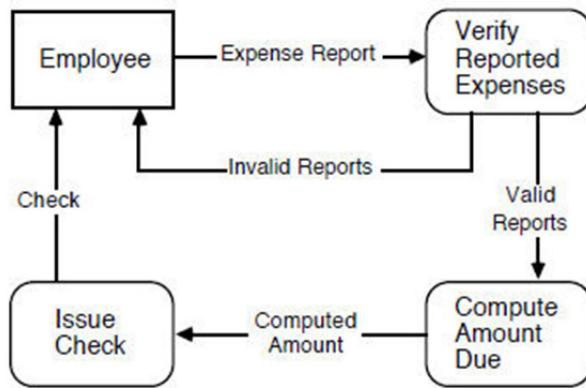


Figure 9 — Data flow diagram

3.1008

data format

1. specified arrangement and encoding for data to be communicated or stored and retrieved
2. arrangement of data for storage or display [ISO/IEC 25024:2015 *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.8]

Note 1 to entry: Format can refer to data type and length of data item.

3.1009

data function

1. functionality provided to the user to meet internal or external data storage requirements [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.16]
2. a logical file [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*]

Note 1 to entry: That is, a logical group of permanent data seen from the perspective of the user. FPA assigns each data function a type and distinguishes between the following types: the internal logical file and the external interface file.

3.1010

data function type

1. one of two categories that FPA assigns to a data function; internal logical file and external interface file [ISO/IEC 24570:2005 *Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1011

data gathering and representation techniques

1. projects to collect, organize and present data and information [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.1012

data group

data group type

1. a distinct, non-empty, non-ordered and non-redundant set of data elements [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, A.5*] 2. a distinct, non-empty, non-ordered and non-redundant set of data attributes where each included data attribute describes a complementary aspect of the same object of interest [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.5*]
cf. object of interest

Note 1 to entry: Each included data element describes a complementary aspect of the same object of interest. A data group is characterized by its persistence.

3.1013

data information

1. information that enters or exits the application and that satisfies the user's information need [*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*]

3.1014

data input sheet

1. user documentation that describes, in a worksheet format, the required and optional input data for a system or component
cf. user manual

3.1015

data inventory

1. in an information processing system, all the data and their characteristics, including interdependencies [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1016

data item

field

1. smallest identifiable unit of data within a certain context for which the definition, identification, permissible values, and other information is specified by means of a set of properties [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.9*]

Note 1 to entry: Data item is a physical object 'container' of data values.

3.1017

data management

1. in a data processing system, the functions that provide access to data, perform or monitor the storage of data, and control input-output operations [*ISO/IEC 2382:2015, Information technology — Vocabulary*] 2. disciplined processes and systems that plan for, acquire, and provide stewardship for business and technical data, consistent with data requirements, throughout the data lifecycle

3.1018

data manipulation

1. any processing of the data other than a movement of the data into or out of a functional process, or between a functional process and persistent storage [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.6*]

3.1019

data medium

1. material in or on which data can be recorded and from which data can be retrieved. Plural: media [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1020

data model

1. graphical and textual representation of analysis that identifies the data needed by an organization to achieve its mission, functions, goals, objectives, and strategies and to manage and rate the organization [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.44]
2. model about data by which an interpretation of the data can be obtained in the modeling tool industry [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]
3. graphical and/or lexical representation of data, specifying their properties, structures, and interrelationships [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.15]

Note 1 to entry: A data model can be encoded and manipulated by a computer. A data model identifies the entities, domains (attributes), and relationships (associations) with other data and provides the conceptual view of the data and the relationships among data [key style]. A distinction is made between a logical (or functional) and a technical data model. A logical data model is a representation of an organization's data, organized in terms of entities and relationships and is independent of any particular data management technology. In a technical data model, it is determined in what form data are recorded in the database and in which way the data are approached.

3.1021

data movement (-type)

1. base functional component which moves a single data group. [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method*, 3.6]

Note 1 to entry: The COSMIC Functional Size Measurement Method has four types of data movements: Entry, Exit, Read and Write. For measurement purposes, each data movement is considered to account for certain associated data manipulation

3.1022

data processing (DP)

automatic data processing (ADP)

1. systematic performance of operations upon data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: arithmetic or logic operations upon data, merging or sorting of data, assembling or compiling of programs, or operations on text, such as editing, sorting, merging, storing, retrieving, displaying, or printing

Note 1 to entry: The term data processing is not a synonym for information processing.

3.1023

data processing system

1. one or more computers, peripheral equipment, and software that perform data processing [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. computer system

3.1024

data protection

1. implementation of appropriate administrative, technical, or physical means to guard against unauthorized intentional or accidental disclosure, modification, or destruction of data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1025

data provider

1. individual or organization that is a source of data [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.5]

3.1026

data quality

1. degree to which the characteristics of data satisfy stated and implied needs when used under specified conditions [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.5; *ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*, 4.1]

3.1027**data quality characteristic**

1. category of data quality attributes that bears on data quality [ISO/IEC 25024:2015 *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.12]

3.1028**data quality measure**

1. variable to which a value is assigned as the result of measurement of a data quality characteristic [ISO/IEC 25024:2015 *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.13]

3.1029**data quality model**

1. defined set of characteristics which provides a framework for specifying data quality requirements and evaluating data quality [ISO/IEC 25024:2015 *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.14]

3.1030**data repository**

1. object providing the storage function [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 14.1.1.1]

3.1031**data store**

1. organized and persistent collection of data and information that allows for its retrieval [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.6]

3.1032**data structure**

1. physical or logical relationship among data elements, designed to support specific data manipulation functions

Note 1 to entry: The data structures are usually documented in technical and logical data models.

3.1033**data structure diagram**

1. diagram that depicts a set of data elements, their attributes, and the logical relationships among them
cf. data flow diagram, entity-relationship diagram

Employee Record									
Emp. No. (4I)	Emp. Name			Emp. Address				Dept. No. (3I)	Emp. Sal. (4I)
	First (10C)	Mid. (1C)	Last (16C)	Street (20C)	City (20C)	State (2C)	Zip (9I)		
I = Integer	C = Character								

Figure 10 — Data structure diagram

3.1034**data structure-centered design**

1. software design technique in which the architecture of a system is derived from analysis of the structure of the data sets with which the system must deal

cf. input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structure clash, structured design, transaction analysis, transform analysis

3.1035

data submitter

- 1.** person or organization that provides IT project data to be included into a benchmarking repository [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*, 3.1]

3.1036

data transfer controller (DTC)

- 1.** functional unit to control data communication without going through the central processing unit (CPU)

3.1037

data type

- 1.** class of data, characterized by the members of the class and the operations that can be applied to them **2.** categorization of an abstract set of possible values, characteristics, and set of operations for an attribute [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.45] **3.** set of values and operations on those values [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.45] **4.** categorization of values operation arguments, typically covering both behavior and representation [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.4]

EXAMPLE: integers, real numbers, and character strings

Note 1 to entry: [key style]

3.1038

data use

- 1.** executable statement where the value of a variable is accessed [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.13]

3.1039

data value

- 1.** content of data item [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.17]

Note 1 to entry: Data quality refers to data itself, such as data domain values and possible restrictions.

3.1040

data-sensitive fault

pattern-sensitive fault

- 1.** fault that causes a failure in response to some particular pattern of data
cf. program-sensitive fault

3.1041

data-structure-oriented design

- 1.** design methodology used for business applications by basing the design on the logical data structures of the program specification

EXAMPLE: Jackson System Design and Warnier-Orr methods

3.1042

database

- 1.** collection of interrelated data stored together in one or more computerized files **2.** collection of data organized according to a conceptual structure describing the characteristics of the data and the relationships among their corresponding entities, supporting one or more application areas [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **3.** collection of data describing a specific target area that is used and updated by one or more applications [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*, A.4]

3.1043

database design specification

- 1.** document that describes the content and format of the permanent or semi-permanent data necessary for the software to carry out its functions

3.1044

database management system

- 1.** organized collection of structured data [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.18*]

Note 1 to entry: In order to use database management systems (DBMS), it is necessary to represent data and the relative operations on it in terms of a data model, a data definition and manipulation language

3.1045

DataItem

- 1.** possible value or structure of values that can be retained or processed [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1046

DataKey

- 1.** identifier for a particular grouping of data [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1047

DataPart

- 1.** component of a structured DataType with a domain specified by reference to some other DataType [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1048

DataRole

- 1.** characterization of the way an entity DataType participates in a relationship [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1049

DataStore

- 1.** specification of data retention capabilities for the subject software [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*]

3.1050

DataType

- 1.** set of possible values or structures of data [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1051

DataView

- 1.** partitioning of a supertype DataType entity into subtype entity [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1052

datum

- 1.** singular of "data"

Note 1 to entry: "Data" is commonly used for both singular and plural.

3.1053

DBMS

- 1.** Database Management System [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 5*]

3.1054

DBR

1. data buffer register

3.1055

DCE

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

3.1056

DDR

1. detailed design review **2.** double data rate

3.1057

DDR2

1. double data rate x 2 (twice as fast as DDR)

3.1058

DDR2 SDRAM

1. double data rate synchronous dynamic random access memory unit with higher performance than DDR SDRAM, because the device transfers data four times (four consecutive words) in one internal clock cycle.

3.1059

DDR3

1. double data rate 3, which transfers data 2 to the third power (8 times) faster than DDR

3.1060

DDR3 SDRAM

1. double data rate synchronous dynamic random access memory unit with higher performance because it transfers data 2 to the third power (8 times) (8 consecutive words) in one internal clock cycle.

3.1061

DDR4

1. double data rate 4; data transfer is 2 to the 4th power = 16 times that of a SDRAM

3.1062

DDR4 SDRAM

1. double data rate synchronous dynamic random access memory unit with higher performance, because it transfers data at the rate of 2 to the 4th power (16) times (16 consecutive words) in one internal clock cycle

3.1063

deactivation

1. checkpointing a cluster, followed by deletion of the cluster [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.23]

3.1064

deadlock

1. situation in which computer processing is suspended because two or more devices or processes are each awaiting resources assigned to the others **2.** situation in which two or more tasks are suspended indefinitely because each task is waiting for a resource acquired by another task

cf. lockout

3.1065

deblock

1. to separate the parts of a block

cf. block (2)

3.1066

debug

1. to detect, locate, and correct faults in a computer program **2.** to detect, locate, and eliminate errors in programs [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

Note 1 to entry: Techniques include use of breakpoints, desk checking, dumps, inspection, reversible execution, single-step operation, and traces.

3.1067

decision

1. types of statements in which a choice between two or more possible outcomes controls which set of actions will result [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.8]

Note 1 to entry: Typical decisions are simple selections (e.g. if-then-else), to decide when to exit loops (e.g. while-loop), and in case (switch) statements (e.g. case-1-2-3-...-N).

3.1068

decision criteria

1. thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.7]

3.1069

decision outcome

1. result of a decision (which therefore determines the control flow alternative taken) [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.14]

3.1070

decision rule

1. combination of conditions (also known as causes) and actions (also known as effects) that produce a specific outcome in decision table testing and cause-effect graphing [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.15]

3.1071

decision table

1. table of all contingencies that are to be considered in the description of a problem together with the action to be taken [*ISO 5806:1984 Information processing — Specification of single-hit decision tables*, 3.1] **2.** table used to show sets of conditions and the actions resulting from them **3.** table of conditions that are to be considered in the analysis of a problem, together with the action to be taken for each condition [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1072

decision tree analysis

1. a diagramming and calculation technique for evaluating the implications of a chain of multiple options in the presence of uncertainty [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1073

decision variable

1. representation of different values for a decision which the decision-maker can choose

Note 1 to entry: for example, in an economic life calculation, the decision variable is how long to keep the asset

3.1074

declaration

1. action that establishes a state of affairs in the environment of the object making the declaration [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language*, 6.6.5] **2.** non-executable program statement that affects the assembler or compiler's interpretation of other statements in the program **3.** set of statements which define the sets, constants, parameter values, typed variables and functions required for defining the annotations on a high-level Petri Net graph [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.8]

Note 1 to entry: The essence of a declaration is that, by virtue of the act of declaration itself and the authority of the object or its principal, it causes a state of affairs to come into existence outside the object making the declaration.

3.1075

declarative language

1. nonprocedural language that permits the user to declare a set of facts and to express queries or problems that use these facts

cf. interactive language, rule-based language

3.1076

decompile

1. to translate a compiled computer program from its machine language version into a form that resembles, but is not necessarily identical to, the original high-order language program

cf. compile

3.1077

decompiler

1. software tool that decompiles computer programs

3.1078

decomposition

1. a technique used for dividing and subdividing the project scope and project deliverables into smaller, more manageable parts [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 2. partitioning of a modeled function into its component functions [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.35*] 3. specification of a given object as a composition [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.3.a*] 4. specification of a given behavior as a composition [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.3.b*]

cf. composition

3.1079

decomposition diagram

1. diagram that details its parent box [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.36*]

3.1080

decoupling

1. process of making software modules more independent of one another to decrease the impact of changes to, and errors in, the individual modules

cf. coupling

3.1081

defect

1. imperfection or deficiency in a work product where that work product does not meet its requirements or specifications and needs to be either repaired or replaced [*IEEE 1044-2009 IEEE Standard Classification for Software Anomalies, 2*] 2. an imperfection or deficiency in a project component where that component does not meet its requirements or specifications and needs to be either repaired or replaced [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 3. generic term that can refer to either a fault (cause) or a failure (effect) [*IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, 2.1*]

cf. fault

EXAMPLE: (1) omissions and imperfections found during early life cycle phases and (2) faults contained in software sufficiently mature for test or operation

3.1082

defect density

1. number of defects per unit of product size

EXAMPLE: problem reports per thousand lines of code

3.1083

defect repair

- 1.** an intentional activity to modify a non-conforming product or product component [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1084

defensive programming

- 1.** general approach to programming that assumes that errors will occur during both initial development and maintenance and, as a result, creates code in such a way that the program still operates properly when errors occur

3.1085

Define Activities

- 1.** the process of identifying the specific actions to be performed to produce the project deliverables [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1086

Define Scope

- 1.** the process of developing a detailed description of the project and product [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1087

defined process

- 1.** implemented process that is managed and tailored from the organization's set of standard processes according to the organization's tailoring guidelines [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.2]

Note 1 to entry: A defined process has a process description that is documented and maintained and contributes work products, measures, and other process improvement information to the organization's process assets. A project's defined process provides a basis for planning, performing, and improving the tasks and activities of the project.

3.1088

definition-use pair

- 1.** data definition and subsequent predicate or computational data use, where the data use uses the value defined in the data definition [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.16]

3.1089

definition-use path

- 1.** data definition and subsequent predicate or computational data use, where the data use uses the value defined in the data definition [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.16]

3.1090

definitive master version

- 1.** originating instance of the software that is used to install the software and to create distribution copies [*ISO/IEC 19770-1:2012 Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3.4]

Note 1 to entry: Install can apply to executable or non-executable software, or related assets such as fonts. It can apply to installs on clients/local devices and/or server-side installs, for example as part of a service type software asset provision.

3.1091

definitive software library (DSL)

- 1.** secure storage environment, formed of physical media or of one or more electronic software repositories, capable of control and protection of definitive authorized versions of all software configuration items and masters of all software controlled by SAM [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.11]

3.1092

degree of influence (DI)

1. a numerical indicator of the impact of each of the 19 (or more) technical complexity adjustment factors, ranging from 0 (no influence) to 5 (strong influence, throughout) [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10]

Note 1 to entry: These indicators are used to compute the value adjustment factor

3.1093

delegation

1. action that assigns something, such as authorization, responsibility or provision of a service, to another object [ISO/IEC 15414:2015 *Information technology — Open distributed processing — Reference model — Enterprise language*, 6.6.6]

Note 1 to entry: A delegation, once made, can later be withdrawn.

3.1094

deleted source statement

1. source statement that is removed or modified from an existing software product as a new product is constructed

3.1095

deletion

1. of an <X>, the action of destroying an instantiated <X> [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.20]

3.1096

delimiter

1. character or set of characters used to denote the beginning or end of a group of related bits, characters, words, or statements

3.1097

deliver primitive

1. service primitive for which the protocol object is the responding object of the corresponding communication [ISO/IEC 14752:2000 *Information technology — Open Distributed Processing — Protocol support for computational interactions*, 3.3.6]

3.1098

deliverable

1. any unique and verifiable product, result, or capability to perform a service that must be produced to complete a process, phase, or project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 2. item to be provided to an acquirer or other designated recipient as specified in an agreement [IEEE 730-2014 *IEEE Standard for Software Quality Assurance Processes*, 3.2]
cf. acquirer, product, result

Note 1 to entry: This item can be a document, hardware item, software item, service, or any type of work product.

3.1099

deliverable product

1. unique and verifiable system or software product to perform a service, that is subject to approval by the project sponsor or customer [ISO/IEC 25041: 2012 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation guide for developers, acquirers and independent evaluators*, 4.1]

3.1100

delivered source statement

1. source statement that is incorporated into the product delivered to the customer

3.1101

delivery

- 1.** release of a system or component to its customer or intended user
cf. software life cycle, system life cycle

3.1102

delphi technique

- 1.** an information gathering technique used as a way to reach a consensus of experts on a subject. Experts on the subject participate in this technique anonymously. A facilitator uses a questionnaire to solicit ideas about the important project points related to the subject. The responses are summarized and are then recirculated to the experts for further comment. Consensus may be reached in a few rounds of this process. The Delphi technique helps reduce bias in the data and keeps any one person from having undue influence on the outcome. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1103

delta

- 1.** difference between two versions [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities*, 3.11]

3.1104

demand paging

- 1.** storage allocation technique in which pages are transferred from auxiliary storage to main storage only when those pages are needed
cf. anticipatory paging

3.1105

DEMIL

- 1.** demilitarization [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.1106

demodularization

- 1.** in software design, the process of combining related software modules, usually to optimize system performance
cf. downward compression, lateral compression, upward compression

3.1107

demonstration

- 1.** dynamic analysis technique that relies on observation of system or component behavior during execution, without need for post-execution analysis, to detect errors, violations of development standards, and other problems
cf. testing

3.1108

demonstrative product

- 1.** product which proves the relevance of a solution [*ISO/IEC TR 14759:1999 Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use*, 3.2 a)]

3.1109

dependability

- 1.** trustworthiness of a computer system such that reliance can be justifiably placed on the service it delivers [*IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*, 2.2] **2.** availability performance and its influencing factors: reliability performance, maintainability performance and maintenance support performance [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.1.4]

3.1110

dependency determination

- 1.** a technique used to identify the type of dependency that is used to create the logical relationships between predecessor and successor activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1111

dependent entity

identifier-dependent entity

1. entity for which the unique identification of an instance depends upon its relationship to another entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.46]

cf. independent entity [key style]

Note 1 to entry: Expressed in terms of the foreign key, an entity is said to be dependent if any foreign key is wholly contained in its primary key.

3.1112

dependent state class

1. class whose instances are, by their very nature, intrinsically related to certain other state class instance(s) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.47]

cf. independent state class

Note 1 to entry: It would not be appropriate to have a dependent state class instance by itself and unrelated to an instance of another class(es) and, furthermore, it makes no sense to change the instance(s) to which it relates.

3.1113

deployment

1. phase of a project in which a system is put into operation and cutover issues are resolved

cf. release

3.1114

deployment package

DP

1. set of artifacts developed to facilitate the implementation of a set of practices of the selected framework in a very small entity [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.8]

3.1115

derived class

1. relation between a template class CA of instances of A, and template class CB of instances of B, where template A is an incremental modification of template B [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.24]

cf. base class

3.1116

derived data

1. data created as a result of processing that involves steps other than or in addition to direct retrieval and validation of information from data functions [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.17] **2.** data that can be derived [*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*]

3.1117

derived measure

1. measure that is defined as a function of two or more values of base measures [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.8]

Note 1 to entry: A transformation of a base measure using a mathematical function can also be considered as a derived measure.

3.1118

derived property

derived attribute

derived participant party

1. designation given to a property whose value is determined by computation [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.50]

Note 1 to entry: The typical case of a derived property is as a derived attribute although there is nothing to prohibit other kinds of derived property.

3.1119

derived requirement

- 1.** requirement deduced or inferred from the collection and organization of requirements into a particular system configuration and solution [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.8] **2.** requirement that is not explicitly stated in customer requirements, but is inferred from contextual requirements (such as applicable standards, laws, policies, common practices, and management decisions) or from requirements needed to specify a product or service component
cf. product requirement

Note 1 to entry: Derived requirements can arise during analysis and design of components of the product or service.

3.1120

derived type

- 1.** data type whose members and operations are taken from those of another data type according to some specified rule
cf. subtype

3.1121

descendent box

- 1.** box in a descendent diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.37]

3.1122

descendent diagram

- 1.** decomposition diagram related to a specific box by a hierarchically consecutive sequence of one or more child/parent relationships [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.38]

3.1123

description

- 1.** information item that represents a planned or actual concept, function, design, or object [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.8]

3.1124

description standard

- 1.** standard that describes the characteristics of product information or procedures provided to help understand, test, install, operate, or maintain the product

3.1125

design

- 1.** [process] to define the architecture, system elements, interfaces, and other characteristics of a system or system element [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.16; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.15] **2.** result of the process in (1) [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.17; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.16] **3.** process of defining the software architecture, components, modules, interfaces, and data for a software system to satisfy specified requirements **4.** process of conceiving, inventing, or contriving a scheme for turning a computer program specification into an operational program **5.** phase of development concerned with determining what documentation will be provided in a product and the nature of the documentation [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.8]
cf. architectural design, preliminary design, detailed design

Note 1 to entry: Design provides the detailed implementation-level physical structure, behavior, temporal relationships, and other attributes of system elements. It is information, including specification of system elements and their relationships, that is sufficiently complete to support a compliant implementation of the architecture.

3.1126

design analyzer

- 1.** automated design tool that accepts information about a program's design and produces such outputs as module hierarchy diagrams, graphical representations of control and data structure, and lists of accessed data blocks

3.1127

design attribute

- 1.** element of a design view that names a characteristic or property of a design entity, design relationship, or design constraint [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.1*]

cf. design constraint, design entity, design relationship

3.1128

design authority

- 1.** person or organization that is responsible for the design of the product [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.5.3; ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.5*]

3.1129

design characteristic

- 1.** design attributes or distinguishing features that pertain to a measurable description of a product or service [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.18*] **2.** design attributes or distinguishing features that pertain to a measurable description of a product or process [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.17*] **3.** design attributes or distinguishing features that pertain to a measurable description of a product or service [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.17*]

3.1130

design concept

- 1.** fundamental idea that can be applied to designing a system

EXAMPLE: information hiding

3.1131

design concern

- 1.** area of interest with respect to a software design [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.2*]

3.1132

design constraint

- 1.** element of a design view that names and specifies a rule or restriction on a design entity, design attribute, or design relationship [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.3*] **2.** explicit and direct restriction regarding the choice of design ideas [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*]

cf. design attribute, design entity, design relationship

EXAMPLE: physical requirements, performance requirements, software development standards, and software quality assurance (SQA) standards

Note 1 to entry: It either declares a design idea to be compulsory or to be excluded.

3.1133

design description

design document

design specification

- 1.** document that describes the design of a system or component

cf. product specification, requirements specification

Note 1 to entry: Typical contents include system or component architecture, control logic, data structures, input/output formats, interface descriptions, and algorithm.

3.1134

design element

- 1.** item occurring in a design view that can be any of the following: design entity, design relationship, design attribute, or design constraint [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.4*]

3.1135

design entity

- 1.** element of a design view that is structurally, functionally, or otherwise distinct from other elements, or plays a different role relative to other design entities [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.5*]

cf. design view

3.1136

design fault

- 1.** design (specification, coding) fault that results from a human error during system design and that might result in a design failure

3.1137

design language

- 1.** specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a hardware or software design **2.** standardized notation, modeling technique, or other representation scheme and its usage conventions, shown to be effective in representing and communicating design information

cf. requirements specification language

Note 1 to entry: Types include hardware design language, program design language.

3.1138

design level

- 1.** design decomposition of the software item

EXAMPLE: system, subsystem, program, or module

3.1139

design methodology

- 1.** systematic approach to creating a design consisting of the ordered application of a specific collection of tools, techniques, and guidelines

3.1140

design of experiments

- 1.** a statistical method for identifying which factors may influence specific variables of a product or process under development or in production [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1141

design overlay

- 1.** representation of additional, detailed, or derived design information organized with reference to an existing design view [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.6*]

cf. design view

3.1142

design pattern

- 1.** description of the problem and the essence of its solution to enable the solution to be reused in different settings

Note 1 to entry: not a detailed specification, but a description of accumulated wisdom and experience.

3.1143

design phase

- 1.** period in the software life cycle during which definitions or designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements
cf. detailed design, preliminary design

3.1144

design rationale

- 1.** information capturing the reasoning of the designer that led to the system as designed, including design options, trade-offs considered, decisions made, and the justifications of those decisions [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.7*]

3.1145

design relationship

- 1.** element of a design view that names a connection or correspondence between design entities [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.8*]
cf. design entity

3.1146

design requirement

- 1.** requirement that specifies or constrains the design of a system or system component
cf. functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement

3.1147

design review

- 1.** formal, documented, comprehensive, and systematic examination of a design to determine if the design meets the applicable requirements, to identify problems, and to propose solutions **2.** process or meeting during which a system, hardware, or software design is presented to project personnel, managers, users, customers, or other interested parties for comment or approval
cf. code review, formal qualification review, requirements review, test readiness review

Note 1 to entry: Types include critical design review, preliminary design review, system design review.

3.1148

design stakeholder

- 1.** individual, organization, or group (or classes thereof playing the same role) having an interest in, or design concerns relative to, the design of some software item [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.9*]
cf. design concern

3.1149

design standard

- 1.** standard that describes the characteristics of a design or a design description of data or program components

3.1150

design strategy

- 1.** overall plan and direction for performing design

EXAMPLE: functional decomposition

3.1151

design subject

software under design
system under design

- 1.** software item or system for which a system design description (SDD) will be prepared [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.10*]

3.1152

design unit

- 1.** logically related collection of design elements

Note 1 to entry: In an Ada PDL, a design unit is represented by an Ada compilation unit.

3.1153

design view

- 1.** representation comprised of one or more design elements to address a set of design concerns from a specified design viewpoint [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.12*]
cf. design concern, design element, design viewpoint

3.1154

design viewpoint

- 1.** specification of the elements and conventions available for constructing and using a design view [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.13*]
cf. design view

3.1155

design-to-cost

cost as an independent variable (CAIV)

- 1.** approach to managing a system or software project so as to hold the project to a predetermined cost
Note 1 to entry: Actual and projected costs are closely tracked, and actions such as deleting or postponing lower-priority requirements are taken if costs threaten to exceed targets.

3.1156

designer

- 1.** stakeholder responsible for devising and documenting the software design [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions, 3.11*]

3.1157

desirable consequence

positive consequence

- 1.** consequence associated with a benefit or gain or avoiding an adverse consequence [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.4.3*]

3.1158

desk checking

- 1.** manual simulation of program execution to detect faults through step-by-step examination of the source program for errors in function or syntax. [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** static analysis technique in which code listings, test results, or other documentation are visually examined, usually by the person who generated them, to identify errors, violations of development standards, or other problems
cf. inspection, walk-through

3.1159

desktop publishing

- 1.** electronic publishing using a microcomputer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1160

destination address

- 1.** address of the device or storage location to which data is to be transferred
cf. source address

3.1161

destructive read

- 1.** read operation that alters the data in the accessed location
cf. nondestructive read

3.1162

DET

- 1.** data element type [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 4*]

3.1163

detailed design

- 1.** process of refining and expanding the preliminary design of a system or component to the extent that the design is sufficiently complete to be implemented **2.** result of the process in (1)
cf. software development process

3.1164

detailed design description

detailed design specification

- 1.** document that describes the exact detailed configuration of a computer program

Note 1 to entry: It identifies the input, output, control logic, algorithms, and data structure of each individual low-level component of the software product and is the primary product of the detailed design phase.

3.1165

detailed design phase

- 1.** software development lifecycle phase during which the detailed design process takes place, using the software system design and software architecture from the previous phase (architectural design) to produce the detailed logic for each unit such that it is ready for coding

3.1166

detailed design review

- 1.** milestone review to determine the acceptability of the detailed software design (as depicted in the detailed design description) to satisfy the requirements of the software requirements document

3.1167

detailed function point count

- 1.** the most accurate count to determine the size of an application or a project in which all the specifications needed for FPA are known in detail [*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*]

Note 1 to entry: This means that transactions have been specified up to the level of referenced logical files (the so-called file types referenced) and data element types, and that logical files have been specified up to the level of record types and data element types. As a result, the complexity of each function recognized can be established.

3.1168

Determine Budget

- 1.** the process of aggregating the estimated costs of individual activities or work packages to establish an authorized cost baseline [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1169

develop project charter

- 1.** the process of developing a document that formally authorizes the existence of a project and provides the project manager with the authority to apply organizational resources to project activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1170

develop project management plan

- 1.** the process of defining, preparing, and coordinating all subsidiary plans and integrating them into a comprehensive project management plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1171

develop project team

- 1.** the process of improving the competencies, team interaction, and the overall team environment to enhance project performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1172

develop schedule

- 1.** the process of analyzing activity sequences, durations, resource requirements, and schedule constraints to create the project schedule model [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1173

developed source statement

- 1.** source statement that is newly created for, added to, or modified for a software product

3.1174

developer

- 1.** individual or organization that performs development activities (including requirements analysis, design, testing through acceptance) during the system or software life-cycle process [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.6; ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.12*] **2.** person who applies a methodology for some specific job, usually an endeavor [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies, 3.11*]

Note 1 to entry: Developers apply methodologies via enactment.

3.1175

development

- 1.** specification, construction, testing and delivery of a new application or of a discrete addition to an existing application [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*] **2.** activity of preparing documentation after it has been designed [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.14*]

EXAMPLE: new development, modification, reuse, reengineering, maintenance, or any other activity that results in software products, and includes the testing, quality assurance, configuration management, and other activities applied to these products.

3.1176

development branch

- 1.** branch where active product development takes place

Note 1 to entry: A product build from the development branch will have the latest features, but will also likely be immature and unstable.

3.1177

development environment

- 1.** hardware, software, platform and tools for designers and developers of computer solutions

3.1178

development plan

- 1.** plan for guiding, implementing, and controlling the design and development of one or more products or services
cf. project plan

3.1179

development project

- 1.** a project in which a completely new application is realized [*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.** project to develop and deliver the first release of a software application [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.18*]

Note 1 to entry: It entails the specification, construction, testing, and delivery of a new application. During actualization, this project can be split up into a number of sub-projects. If these are carried out more or less in parallel, each being responsible for effectuating a certain sub-system of the total application, then each sub-project can be considered as an individual

development project, if the sub-system itself is an application. Re-building an existing application, otherwise known as re-engineering, is considered as development.

3.1180

development project function point count (DFP)

- 1.** a count that measures a project that provides end-users with the first installation of the software [*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 count that measures the functionality provided to the end users with the first installation of the software, developed when the project is complete [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*] **3.** activity of applying ISO/IEC 20926:2009 to measure the functional size of a development project [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.20*]

3.1181

development project functional size

- 1.** measure of the functionality provided to the users with the first release of the software, as measured by the development project function point count [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.19*]

Note 1 to entry: The functional size of a development project can include the size of conversion functionality.

3.1182

development testing

- 1.** formal or informal testing conducted during the development of a system or component, usually in the development environment by the developer. **2.** testing conducted to establish whether a new software product or software-based system (or components of it) satisfies its criteria.

cf. acceptance testing, operational testing, qualification testing

Note 1 to entry: The criteria will vary based on the level of test being performed.

3.1183

development tool

- 1.** hardware and software for developing or modifying applications

3.1184

developmental baseline

- 1.** specifications that are in effect at a given time for a system under development [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1185

developmental configuration

- 1.** in configuration management, the software and associated technical documentation that define the evolving configuration of a computer software configuration item during development
cf. allocated baseline, functional baseline, product baseline

Note 1 to entry: The developmental configuration is under the developer's control, and therefore is not called a baseline.

3.1186

deviation

- 1.** departure from a specified requirement **2.** written authorization, granted prior to the manufacture of an item, to depart from a particular performance or design requirement for a specific number of units or a specific period of time

3.1187

device

- 1.** mechanism or piece of equipment designed to serve a purpose or perform a function
cf. platform

3.1188

device interface task

- 1.** concurrent task that hides the characteristics of and interfaces to an external I/O device

3.1189

DFD

1. data flow diagram

3.1190

diagnostic

1. pertaining to the detection and isolation of faults or failures

EXAMPLE: a diagnostic message, a diagnostic manual

3.1191

diagnostic manual

1. document that presents the information necessary to execute diagnostic procedures for a system or component, identify malfunctions, and remedy those malfunctions

cf. installation manual, operator manual, programmer manual, support manual, user manual

Note 1 to entry: Typically described are the diagnostic features of the system or component and the diagnostic tools available for its support.

3.1192

diagonal microinstruction

1. microinstruction capable of specifying a limited number of simultaneous operations needed to carry out a machine language instruction

cf. horizontal microinstruction, vertical microinstruction

Note 1 to entry: Diagonal microinstructions fall, in size and functionality, between horizontal microinstructions and vertical microinstructions. The designation 'diagonal' refers to this compromise rather than to any physical characteristic of the microinstruction.

3.1193

diagram

1. logically coherent fragment of a design view, using selected graphical icons and conventions for visual representation from an associated design language, to be used for representing selected design elements of interest for a system under design from a single viewpoint (diagram type) [*IEEE 1016-2009 IEEE standard for Information Technology-Systems Design-Software Design Descriptions*, 3.14] **2.** instantiation of the formal diagram structure that consists only of semantically and syntactically valid IDEF0 graphical statements [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.39]
cf. design subject

Note 1 to entry: Each diagram is a single unit of an IDEF0 model that presents the top-level function that is the subject of the model (the A-0 context diagram), presents the context of the subject function (other context diagrams), or presents the details of a box (decomposition diagrams).

3.1194

diagram boundary

1. edge of a diagram in a diagram page [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.40]

3.1195

diagram feature

1. element of a diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.41]

Note 1 to entry: Diagram features include boxes, arrow segments, arrow labels, ICOM codes, ICOM labels, model notes, and reader notes.

3.1196

diagram feature reference

1. expression that unambiguously identifies a diagram feature within an IDEF0 model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.42]

3.1197

diagram number

- 1.** that part of a diagram reference that corresponds to a diagram's parent function's node number [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.43*]

Note 1 to entry: The diagram number refers to the diagram that details or decomposes the function designated by the same node number.

3.1198

diagram page

- 1.** model page that contains a context diagram or a decomposition diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.44*]

3.1199

diagram reference

- 1.** expression that unambiguously identifies a diagram and specifies the diagram's position in a specific model hierarchy [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.45*]

Note 1 to entry: A diagram reference is composed of a model name, abbreviation and a diagram number.

3.1200

diagram title

- 1.** verb or verb phrase that describes the overall function presented by a diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.46*]

Note 1 to entry: The diagram title of a child diagram is the box name of its parent box.

3.1201

diagramming techniques

- 1.** approaches to presenting information with logical linkages that aid in understanding [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1202

dialog

- 1.** conversation between the user and the application needed to execute a transaction [*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.** interaction between a user and an interactive system as a sequence of user actions (inputs) and system responses (outputs) in order to achieve a goal [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.4*]

3.1203

DIB

- 1.** Directory Information Base [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service, 4*]

3.1204

differential cash flow

- 1.** representation of the difference between cash flows of two alternatives or proposals
cf. incremental analysis

Note 1 to entry: often performed using internal rate of return (IRR) as the basis of comparison

3.1205

digit numeric character

numeric character

- 1.** character that represents a nonnegative integer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: one of the characters 0, 1 ... , F in the hexadecimal numeration system

3.1206

digital

1. pertaining to data that consists of digits as well as to processes and functional units that use the data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1207

digital computer

1. computer that is controlled by internally stored programs and that is capable of using common storage for all or part of a program and also for all or part of the data necessary for the execution of the programs; executing user-written or user-designated programs; performing user-designated manipulation of digitally represented discrete data, including arithmetic operations and logic operations; and executing programs that modify themselves during their execution [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1208

digital signal processing (DSP)

1. modification of an information signal represented by a sequence of digits or symbols to affect the representation of discrete time, discrete frequency, or other attributes

3.1209

digital signal processor (DSP)

1. microprocessor designed to perform digital signal processing

3.1210

DII

1. dynamic invocation interface [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.1211

dimension

1. distinct components that a multidimensional construct encompasses [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks*, 3.6]

3.1212

DIP

1. dual inline package

3.1213

direct address

one-level address

1. address that identifies the storage location of an operand

cf. immediate data, indirect address, n-level address, direct instruction

3.1214

direct and manage project work

1. the process of leading and performing the work defined in the project management plan and implementing approved changes to achieve the project's objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1215

direct instruction

1. computer instruction that contains the direct addresses of its operands

cf. immediate instruction, indirect instruction, absolute instruction, effective instruction

3.1216

direct labor

1. personnel efforts that are directly related to the units of production

cf. indirect labor

3.1217

direct measure

1. measure of an attribute that does not depend upon a measure of any other attribute

3.1218

direct memory access (DMA)

1. technique in which a peripheral takes direct control of a central processing unit's memory bus to transfer data to or from memory

3.1219

direct memory access controller (DMAC)

1. functional unit that performs direct memory access

3.1220

direct metric

1. metric that does not depend upon a measure of any other attribute [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.4*]

3.1221

direct metric value

1. numerical target for a quality factor to be met in the final product [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.5*]

EXAMPLE: Mean time to failure (MTTF) is a direct metric of final system reliability.

3.1222

direct staff-hour

1. amount of effort directly expended in creating a specific output product

3.1223

directed graph

digraph

1. graph in which direction is implied in the internode connections

cf. undirected graph

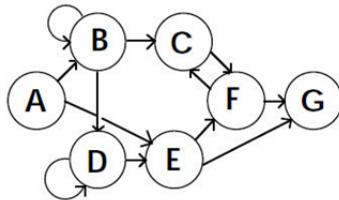


Figure 11 — Directed graph

3.1224

directory

1. list of data items and information about those data items

3.1225

disassemble

1. to translate an assembled computer program from its machine language version into a form that resembles, but is not necessarily identical to, the original assembly language program
cf. assemble

3.1226

disassembler

1. software tool that disassembles computer programs

3.1227

disaster recovery

1. in computer system operations, the return to normal operation after a hardware or software failure

3.1228

disclaimer

1. notice that renounces or repudiates a legal claim or right

3.1229

discounted payback period

- 1.** time it will take to recover a project's initial investment including interest

Note 1 to entry: An indication of exposure to risk. If a project is canceled before it reaches its payback period, the organization will have lost money.

3.1230

discrete

- 1.** pertaining to data that consist of distinct elements, such as characters, or to physical quantities having a finite number of distinctly recognizable values, as well as to processes and functional units that use those data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1231

discrete data

- 1.** data that arrives at specific time intervals

3.1232

discrete effort

- 1.** work effort that is separate, distinct, and related to the completion of specific work breakdown structure components and deliverables, and that can be directly planned and measured **2.** an activity that can be planned and measured and that yields a specific output [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. apportioned effort

Note 1 to entry: One of three earned value management types of activities used to measure work performance

3.1233

discrete type

- 1.** data type whose members can assume any of a set of distinct values

Note 1 to entry: A discrete type can be an enumeration type or an integer type.

3.1234

discretionary dependency

preferential logic

preferred logic

soft logic

- 1.** a relationship that is established based on knowledge of best practices within a particular application area or an aspect of the project where a specific sequence is desired [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1235

discrimination (threshold)

- 1.** largest change in a stimulus that produces no detectable change in the response of a measuring instrument, the change in the stimulus taking place slowly and monotonically [*ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*, 3.4]

Note 1 to entry: The discrimination threshold can depend on, for example, noise (internal or external) or friction. It can also depend on the value of the stimulus.

3.1236

discriminator

:category discriminator

- 1.** property of a superclass, associated with a cluster of that superclass, whose value identifies to which subclass a specific instance belongs [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.51] **2.** attribute in the generic entity (or a generic ancestor entity) of a category cluster whose values indicate which category entity in the category cluster contains a specific instance of the generic entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.51] *Syn*

Note 1 to entry: Since the value of the discriminator (when a discriminator has been declared) is equivalent to the identity of the subclass to which the instance belongs, there is no requirement for a discriminator in identity-style modeling.

3.1237

disk

1. data medium originally consisting of a flat circular plate that is rotated in order to read or write data on one or both sides [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.1238

display

1. information presented on a screen or in a window of a screen [ISO/IEC 26513:2009 *Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.10]

3.1239

disposed system

1. system that has been transformed (i.e. state change) by applying the disposal process [ISO/IEC TR 29110-5-6-2:2014 *Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile*, 3.3]

Note 1 to entry: A systems approach considers the total system and the total lifecycle of the system. This includes all aspects of the system and the system throughout its life until the day users dispose of the system and the external enterprises complete the handling of the disposed system products.

3.1240

distributed computing

1. spreading of computation and data across a number of computers connected by a network

3.1241

distributed processing

1. information processing in which discrete components can be located in different places, and where communication between components can suffer delay or fail [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 3.2.2]

3.1242

distribution copy

1. copy of the software definitive master version, for the purposes of installation onto other hardware, which resides, for example, on a server or on physical media such as CDs [ISO/IEC 19770-1:2012 *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3. 5]

3.1243

distribution transparency

1. property of hiding from the user some specific aspects of the system's complexity needed to support distribution [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 11.1.1]

3.1244

disturbance

1. operational fault or event or anything that could change the state of the system [ISO/IEC 25045:2010 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability*, 4.2]

EXAMPLE: an abrupt shutdown of an OS process that brings down a system, a significant increase in users of the system

Note 1 to entry: Disturbances are limited to external faults or events, rather than introduced internal faults that would require modifying the application or OS code.

3.1245

DIT

1. Directory Information Tree [ISO/IEC 13235-3:1998 *Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service*, 4]

3.1246

diversity

- 1.** in fault tolerance, realization of the same function by different means
cf. software diversity

EXAMPLE: use of different processors, storage media, programming languages, algorithms, or development teams

3.1247

dividing action

- 1.** action which enables two or more chains [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.1.4]

3.1248

DL

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

3.1249

DLC

- 1.** Data-Life-Cycle [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 5]

3.1250

DMA

- 1.** direct memory access

3.1251

DMAC

- 1.** direct memory access controller

3.1252

DN

- 1.** Distinguished Name [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service*, 4]

3.1253

DNS

- 1.** Domain Name Service [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.1254

do nothing alternative

- 1.** in a decision analysis, the alternative of not investing in any of the proposed alternatives

Note 1 to entry: doesn't really mean doing nothing at all. Instead, it means putting the money into readily available investments that give a predetermined rate of return (bonds, interest bearing accounts, put into a more profitable part of the organization)

3.1255

document

- 1.** uniquely identified unit of information for human use, such as a report, specification, manual or book, in printed or electronic form [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.9] **2.** to create a document as in (1) **3.** to add comments to a computer program **4.** separately identified piece of documentation which could be part of a documentation set [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.15] **5.** medium, and the information recorded on it, that generally has permanence and can be read by a person or a machine.

EXAMPLE: policies, plans, process descriptions, procedures, service level agreements, contracts or records. In software engineering: project plans, specifications, test plans, user manuals

Note 1 to entry: Documents include both paper and electronic documents. The documentation can be in any form or type of medium. Documents, except for records, state the intent to be achieved.

3.1256

document analysis

- 1.** an elicitation technique that analyzes existing documentation and identifies information relevant to the requirements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1257

document control

- 1.** application of configuration management to the control of documents

3.1258

document set

- 1.** collection of documentation that has been segmented into separately identified volumes or files for ease of distribution or use [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.17*]

3.1259

documentation

- 1.** collection of documents on a given subject **2.** written or pictorial information describing, defining, specifying, reporting, or certifying activities, requirements, procedures, or results **3.** process of generating or revising a document **4.** information that explains how to use a software product [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.16*] **5.** management of documents, including identification, acquisition, processing, storage, and dissemination.

EXAMPLE: printed manuals, on-screen information, and stand-alone online help

Note 1 to entry: can be provided as separate documentation or as embedded documentation or both

3.1260

documentation plan

- 1.** plan identifying the documents to be produced during the system or software life cycle [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.10*]

3.1261

documentation reviews

- 1.** the process of gathering a corpus of information and reviewing it to determine accuracy and completeness [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1262

documentation tree

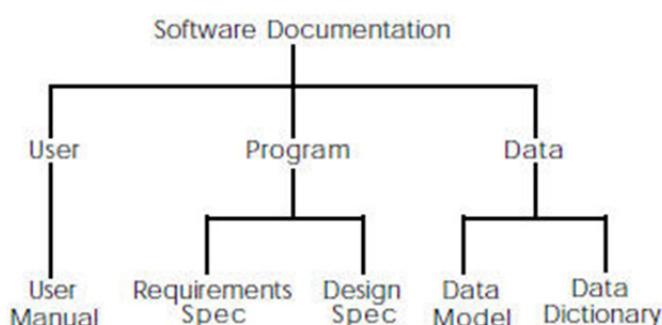


Figure 12 — Documentation tree

3.1263

DODAF

1. Department of Defense architecture framework [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.1264

domain

1. distinct scope, within which common characteristics are exhibited, common rules observed, and over which a distribution transparency is preserved [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.5] **2.** problem space [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1; *IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes*, 3] **3.** area of knowledge or activity characterized by a set of concepts and terminology understood by practitioners in that area [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.1265

domain analysis

1. analysis of systems within a domain to discover commonalities and differences among them [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1] **2.** process by which information used in developing software systems is identified, captured, and organized so that it can be reused to create new systems, within a domain. [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1] **3.** result of the domain analysis process. [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1]

3.1266

domain architecture

product line architecture

reference architecture

1. generic, organizational structure or design for software systems in a domain. [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes*, 3] **2.** core architecture that captures the high-level design of a software and systems product line including the architectural structure and texture (e.g. common rules and constraints) that constrains all member products within a software and systems product line [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.10]

Note 1 to entry: The domain architecture contains the designs that are intended to satisfy requirements specified in the domain model. The domain architecture documents design, whereas the domain model documents requirements. A domain architecture: 1) can be adapted to create designs for software systems within a domain, and 2) provides a framework for configuring assets within individual software systems. The term "architecture" has been deliberately redefined to more properly convey its meaning in the software reuse context.

3.1267

domain asset

core asset

1. output of domain engineering life cycle processes that can be reused in producing products during application engineering [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.11]

EXAMPLE: domain features, domain models, domain requirements specification, domain architecture, domain components, domain test cases, domain process description

3.1268

domain assets in requirements

1. reusable artifacts produced during domain requirements engineering [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering*, 3.10]

EXAMPLE: asset proposals, domain requirements specifications, domain requirements models

3.1269

domain engineer

1. party that performs domain engineering activities, including domain analysis, domain design, asset construction, and asset maintenance [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]

3.1270

domain engineering

1. reuse-based approach to defining the scope (i.e., domain definition), specifying the structure (i.e., domain architecture), and building the assets for a class of systems, subsystems, or applications [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*] 2. life cycle consisting of a set of processes for specifying and managing the commonality and variability of a product line [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.12*]

Note 1 to entry: For example, "assets" such as requirements, designs, software code, documentation. Domain engineering can include the following activities: domain definition, domain analysis, developing the domain architecture, and domain implementation.

3.1271

domain engineering process

1. processes for domain asset development [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management, 3.5*]

3.1272

domain expert

1. individual who is intimately familiar with the domain and can provide detailed information to the domain engineers [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]

3.1273

domain model

1. product of domain analysis that provides a representation of the requirements of the domain [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]

Note 1 to entry: The domain model identifies and describes the structure of data, flow of information, functions, constraints, and controls within the domain that are included in software systems in the domain. The domain model describes the commonalities and variabilities among requirements for software systems in the domain.

3.1274

domain requirements analysis

1. subprocess that models domain requirements so as to analyze and scrutinize commonality/variability of a product line in requirements [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.12*]

3.1275

domain requirements elicitation

1. subprocess that identifies initial requirements from domain stakeholders for a product line [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.11*]

3.1276

domain requirements management

1. subprocess that manages traceability and changes with respect to domain requirements and their relevant domain/application artefacts [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.15*]

3.1277

domain requirements specification

1. subprocess that documents domain requirements for a product line based on domain analysis results [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.13*]

3.1278

domain requirements verification and validation

1. subprocess that confirms that domain requirements are correct, consistent, and complete [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.14*]

3.1279

domain scoping

1. subprocess for identifying and bounding the functional domains that are important to an envisioned product line and provide sufficient reuse potential to justify the product line creation [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.13*]

3.1280

dominance

1. decision technique that looks for an alternative that is at least as good in every attribute and better in at least one attribute

cf. lexicography, satisficing

3.1281

dot notation

1. technique for naming that joins the name of a parent class to the name of a dependent class with the period character [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.47*]

EXAMPLE: The diagram feature reference ABC/A31.3 uses dot notation to join the page reference of the parent diagram ABC/A31 to the feature reference for box 3 in that diagram.

3.1282

double data rate (DDR) SDRAM

1. synchronous dynamic random access memory unit with higher access speed and bandwidth, because it transfers two consecutive words in one internal clock cycle

3.1283

down

1. pertaining to a system or component that is not operational or has been taken out of service
cf. up, busy, crash, idle

3.1284

down time

1. period of time during which a system or component is not operational or has been taken out of service
cf. up time, busy time, idle time, mean time to repair, set-up time

3.1285

downgrade right

1. right granted to receive, install, or use an installation of a previous version of software than the currently granted entitlement [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.7*]

3.1286

download

1. to transfer programs or data from a computer to a connected computer with fewer resources [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: typically, from a server to a personal computer

3.1287

downward compatible

1. pertaining to hardware or software that is compatible with an earlier or less complex version of itself
cf. upward compatible

EXAMPLE: a program that handles files created by an earlier version of itself

3.1288

downward compression

1. in software design, a form of demodularization in which a superordinate module is copied into the body of a subordinate module

cf. lateral compression, upward compression

DP

1. data processing [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** deployment package [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 4.2]

3.1289

DR

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

3.1290

DRAM

1. dynamic random access memory

3.1291

driver

1. software module that invokes and, perhaps, controls and monitors the execution of one or more other software modules **2.** computer program that controls a peripheral device and, sometimes, reformats data for transfer to and from the device

cf. test driver

3.1292

DSA

1. Directory System Agent [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service*, 4]

3.1293

DSL

1. definitive software library [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.11]

3.1294

DSP

1. digital signal processing **2.** digital signal processor [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.1295

DT

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

3.1296

DT&E

1. developmental test and evaluation [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.1297

DTC

1. data transfer controller

3.1298

DTD

1. document type definition [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

Note 1 to entry: for XML or SGML specifications

3.1299

DUA

1. Directory User Agent [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service, 4*]

3.1300

dual boot

1. having more than one boot mode, to allow running two different operating systems on the same computer
cf. single boot

3.1301

dual inline package (DIP)

1. microcircuit unit with connectors (pins) arranged in two rows

3.1302

dumb terminal

nonprogrammable terminal

1. user terminal that has no independent data processing capability [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1303

dump

1. display of some aspect of a computer program's execution state, usually the contents of internal storage or registers **2.** display of the contents of a file or device **3.** to copy the contents of internal storage to an external medium **4.** to produce a display or copy as in (1), (2), or (3)

Note 1 to entry: Types include change dump, dynamic dump, memory dump, postmortem dump, selective dump, snapshot dump, static dump.

3.1304

duration (DU or DUR)

1. the total number of work periods (not including holidays or other nonworking periods) required to complete a schedule activity or work breakdown structure component. Usually expressed as workdays or workweeks. Sometimes incorrectly equated with elapsed time [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. effort

3.1305

DVD

1. digital versatile disc [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.2*] **2.** digital video disc

3.1306

dyadic selective construct

1. if-then-else construct in which processing is specified for both outcomes of the branch
cf. monadic selective construct

3.1307

dynamic

1. pertaining to an event or process that occurs during computer program execution
cf. static

EXAMPLE: dynamic analysis, dynamic binding

3.1308

dynamic analysis

1. process of evaluating a system or component based on its behavior during execution
cf. static analysis demonstration, testing

3.1309

dynamic binding

1. binding performed during the execution of a computer program

cf. static binding

3.1310

dynamic breakpoint

1. breakpoint whose predefined initiation event is a runtime characteristic of the program, such as the execution of any twenty source statements

cf. static breakpoint, code breakpoint, data breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint

3.1311

dynamic buffering

1. buffering technique in which the buffer allocated to a computer program varies during program execution, based on current need

cf. simple buffering

3.1312

dynamic bus sizing

1. capability to adjust the size of a bus on request during operations

Note 1 to entry: used during direct memory access

3.1313

dynamic dump

1. dump that is produced during the execution of a computer program

cf. static dump, change dump, memory dump, postmortem dump, selective dump, snapshot dump

3.1314

dynamic error

1. error that is dependent on the time-varying nature of an input

cf. static error

3.1315

dynamic invocation

1. constructing and issuing a request whose signature is possibly not known until run-time [ISO/IEC 19500-2:2012 *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.6]

3.1316

dynamic model

1. model that describes individual requests or patterns of requests among objects [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.53]
cf. static model

3.1317

dynamic product

1. system or software product that is measurable during execution in a testing or an operational environment [ISO/IEC 25041: 2012 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation guide for developers, acquirers and independent evaluators*, 4.2]

3.1318

dynamic random access memory (DRAM)

1. RAM with a frequent refresh process to retain data

Note 1 to entry: used with a circuit architecture in single stable state

3.1319

dynamic relocation

1. relocation of a computer program during its execution

3.1320

dynamic resource allocation

- 1.** computer resource allocation technique in which the resources assigned to a program vary during program execution, based on current need

3.1321

dynamic restructuring

- 1.** process of restructuring a database, data structure, computer program, or set of system components during program execution

3.1322

dynamic schema

- 1.** specification of the allowable state changes of one or more information objects, subject to the constraints of any invariant schemata [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 6.1.3]

Note 1 to entry: Behavior in an information system can be modeled as transitions from one static schema to another, i.e., reclassification of instances from one type to another. In the information language, a state change involving a set of objects can be regarded as an interaction between those objects. Not all of the objects involved in the interaction need change state; some of the objects can be involved in a read-only manner.

3.1323

dynamic skeleton

- 1.** interface-independent kind of skeleton, used by servers to handle requests whose signatures are possibly not known until run-time [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.7]

3.1324

dynamic storage allocation

- 1.** storage allocation technique in which the storage assigned to a computer program varies during program execution, based on the current needs of the program and of other executing programs

3.1325

dynamic testing

- 1.** testing that requires the execution of the test item [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.9] **2.** testing that requires the execution of program code [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes*, 4.4]

3.1326

E-R diagram

- 1.** entity-relationship diagram

3.1327

EAC

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

3.1328

early finish date (EF)

- 1.** in the critical path method, the earliest possible point in time on which the uncompleted portions of a schedule activity can finish, based on the schedule network logic, the data date, and any schedule constraints [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1329

early start date (ES)

- 1.** in the critical path method, the earliest possible point in time on which the uncompleted portions of a schedule activity can start, based on the schedule network logic, the data date, and any schedule constraints [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1330

early-failure period

burn-in period

1. period of time in the life cycle of a system or component during which hardware failures occur at a decreasing rate as problems are detected and repaired

cf. constant-failure period, wearout-failure period, bathtub curve

3.1331

earned value (EV)

budgeted cost of work performed (BCWP)

1. the measure of work performed expressed in terms of the budget authorized for that work [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1332

earned value management (EVM)

1. a methodology that combines scope, schedule, and resource measurements to assess project performance and progress [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1333

echo

1. to return a transmitted signal to its source, often with a delay to indicate that the signal is a reflection rather than the original **2.** returned signal

3.1334

economic risk mitigation

1. degree to which a product or system mitigates the potential risk to financial status, efficient operation, commercial property, reputation, or other resources in the intended contexts of use [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.1.4.1*]

3.1335

ECP

1. engineering change proposal [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.1336

ECR

1. engineering change request [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.1337

EDA

1. electronic design automation

3.1338

EDI

1. electronic data interchange

3.1339

edit

1. to modify the form or format of computer code, data, or documentation

EXAMPLE: to insert, rearrange, or delete characters

3.1340

EDRAP

1. engineering data requirements agreement plan [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.1341

EEPROM

- 1.** electric erasable programmable read only memory

3.1342

EF

- 1.** early finish date [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1343

effective address

- 1.** address that results from performing any required indexing, indirect addressing, or other address modification on a specified address

cf. generated address, indirect address, relative address

Note 1 to entry: If the specified address requires no modification, it is also the effective address.

3.1344

effective full license

- 1.** license rights for software which allow one full use of the software [*ISO/IEC 19770-1:2012 Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance, 3.6*]

EXAMPLE: An underlying full license for version 1 of a software product, plus an underlying upgrade license to version 2 of the software product, combine to produce one effective full license for version 2 of the software product.

Note 1 to entry: An effective license consists of one or more underlying licenses. Full use of the software is as defined in the terms and conditions of the license(s).

3.1345

effective instruction

- 1.** computer instruction that results from performing any required indexing, indirect addressing, or other modification on the addresses in a specified computer instruction

cf. absolute instruction, direct instruction, immediate instruction, indirect instruction

Note 1 to entry: If the specified instruction requires no modification, it is also the effective instruction.

3.1346

effective interest rate

- 1.** interest rate that has been adjusted for more or less frequent compounding

3.1347

effectiveness

- 1.** accuracy and completeness with which users achieve specified goals [*ISO/IEC 25062:2006 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports, 4.2*] **1.** extent to which planned activities are realized and planned results achieved [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.1.3*] **2.** relation of the goals of using the product to the accuracy and completeness with which these goals might be achieved [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation, 3.14*]

EXAMPLE: For documentation, common measures include percentage of task completion, frequency of defects, frequency of assists, frequency of accesses to help or documentation.

3.1348

efferent

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

cf. afferent

3.1349

efficiency

1. degree to which a system or component performs its designated functions with minimum consumption of resources 2. resources expended in relation to the accuracy and completeness with which users achieve goals [ISO/IEC 25062:2006 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports*, 4.3] 3. relation of the level of effectiveness achieved to the quantity of resources expended [ISO/IEC 26513:2009 *Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.15] 4. relationship between the result achieved and the resources used [ISO/IEC 16350-2015 *Information technology — Systems and software engineering — Application management*, 4.17]

cf. execution efficiency, storage efficiency

Note 1 to entry: Time-on-task and Completion Rate/Mean Time-On-Task (defect rates vs. time to achieve task) are measures of efficiency. Efficiency is the degree to which an information system efficiently uses the technical infrastructure and thus becomes useable for the customer. The most important underlying topic here is the capacity of the platform in relation to the demand.

3.1350

effort

1. the number of labor units required to complete a schedule activity or work breakdown structure component, often expressed in hours, days or weeks [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. duration

3.1351

egoless programming

1. software development technique based on the concept of team, rather than individual, responsibility for program development

Note 1 to entry: Its purpose is to prevent individual programmers from identifying so closely with their work that objective evaluation is impaired.

3.1352

EI

1. external input [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 4; 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*]

3.1353

EIF

1. external interface file [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 4; 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*]

3.1354

EJB

1. Enterprise Java Beans [ISO/IEC 19500-3:2012 *Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.1355

electric erasable programmable read only memory (EEPROM)

1. type of programmable ROM in which the memory can be erased using electrical current and rewritten cf. flash memory

3.1356

electronic data interchange (EDI)

1. structured way of transmitting data held electronically from database to database, usually using telecommunications networks

3.1357

electronic design automation (EDA)

- 1.** software-driven design and development of electronic components such as microcomputer units and circuit boards

3.1358

electronic mail (Email)

- 1.** correspondence in the form of messages transmitted over a computer network [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1359

electronic publishing

- 1.** production of typeset-quality documents including text, graphics, and pictures with the assistance of a computer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1360

element

- 1.** [system] identifiable part **2.** component of an information structure that provides information related to the entity represented by the information structure [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.12] **3.** one of the parts of a compound or complex whole [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4] **4.** smaller part of an architecture [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.19] **5.** component of an XML document or part of the entitlement schema (Ent) that provides information related to the entitlement represented by the Ent [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.8]
cf. component, unit

EXAMPLE: documents, requirements specifications, test cases, source code, installation information, and read-me files

3.1361

element type

- 1.** category or class of elements

3.1362

elementary process

- 1.** smallest unit of activity that is meaningful to the user [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.21]

3.1363

ELSE-rule

- 1.** actions to be taken for all combinations of conditions not covered by the other rules in the table [*ISO 5806:1984 Information processing — Specification of single-hit decision tables*, 3.5]

Note 1 to entry: The use of the ELSE-rule facility is optional.

3.1364

embedded computer system

embedded system

- 1.** computer system that is part of a larger system and performs some of the requirements of that system

EXAMPLE: a computer system used in an aircraft or rapid transit system

Note 1 to entry: The hardware and software of an embedded system are usually minimized and optimized for specific functions. The embedded system includes at least one microcontroller, microprocessor or digital signal processor. The embedded system designed to optimize reliability, cost, size and power saving for applications.

3.1365

embedded documentation

- 1.** information that is delivered as an integral part of a piece of software [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.16]
cf. separate documentation

EXAMPLE: on-screen help provided with the software

3.1366

embedded middleware

- 1.** software that communicates between an embedded operating system and an embedded application or firmware

3.1367

embedded operating system

- 1.** operating system software for an embedded computer system

3.1368

embedded software

- 1.** software that is part of a larger system and performs some of the requirements of that system

EXAMPLE: software used in an aircraft or rapid transit system

3.1369

EMC

- 1.** electromagnetic compatibility [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.1370

EMD

- 1.** engineering and manufacturing development [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.1371

emergency maintenance

- 1.** unscheduled modification performed to temporarily keep a system operational pending corrective maintenance [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance*, 3.3]

Note 1 to entry: Emergency maintenance is a part of corrective maintenance.

3.1372

EMI

- 1.** electromagnetic interference [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.1373

emitter

- 1.** event source that can be connected to at most one consumer [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

3.1374

emotional intelligence

- 1.** the capability to identify, assess, and manage the personal emotions of oneself and other people, as well as the collective emotions of groups of people [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1375

emulated user

1. imitation of a user, with regard to the tasks he submits and his time behavior, realized by a technical system [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.6*]

3.1376

emulation

1. model that accepts the same inputs and produces the same outputs as a given system **2.** process of developing or using a model **3.** the use of a data processing system to imitate another data processing system, so that the imitating system accepts the same data, executes the same programs, and achieves the same results as the imitated system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. simulation

3.1377

emulator

1. device, computer program, or system that accepts the same inputs and produces the same outputs as a given system

cf. simulator

Note 1 to entry: often used for testing or debugging

3.1378

EMV

1. expected monetary value [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1379

enabled behavior

1. behavior characterizing a set of objects which becomes possible as a result of establishing behavior [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.2.2*]

3.1380

enabling (a transition)

1. transition is enabled in a particular mode and net marking, when the following conditions are met: (1) the marking of each input place of the transition satisfies the demand imposed on it by its arc annotation evaluated for the particular transition mode; (2) the demand is satisfied when the place's marking contains (at least) the multiset of tokens indicated by the evaluated arc annotation [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.9*]

Note 1 to entry: The determination of transition modes guarantees that the transition condition is satisfied.

3.1381

enabling system

1. system that supports a system-of-interest during its life cycle stages but does not necessarily contribute directly to its function during operation [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.19; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.18*]

Note 1 to entry: For example, when a system-of-interest enters the production stage, an enabling production system is required. Each enabling system has a life cycle of its own.

3.1382

enabling tokens

1. multiset of values obtained when an input arc annotation is evaluated for a particular binding to variables [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.25.1*]

3.1383

enactment

1. establishment of something by law, ruling, or other authoritative acts **2.** act of applying a methodology for some particular purpose, typically an endeavor [ISO/IEC 24744:2014 *Software Engineering — Metamodel for development methodologies*, 3.8]

3.1384

encapsulation

1. software development technique that consists of isolating a system function or a set of data and operations on those data within a module and providing precise specifications for the module **2.** concept that access to the names, meanings, and values of the responsibilities of a class is entirely separated from access to their realization [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.54] **3.** the idea that a module has an outside that is distinct from its inside, that it has an external interface and an internal implementation
cf. data abstraction, information hiding

3.1385

encoding

1. definition of how the elements of a syntax are represented using an identified character set [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: Details of representation of the various terminal symbols and data types in the syntax's grammar are provided.

3.1386

ENCODING.1

1. primary encoding defined within the CDIF family of standards [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: The CDIF family of standards supports multiple transfer formats, each composed of a syntax and an encoding.

3.1387

end item

1. entity that is ready for use

3.1388

end of period convention

1. representation of discrete cash-flow instances at the end of the period in which they occur (in contrast to showing them at the beginning)

Note 1 to entry: The initial investment is shown at the end of period zero.

3.1389

end user

1. person who directly uses the system for its intended purpose **2.** individual person who ultimately benefits from the outcomes of the system or software [ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.7] **3.** any person that communicates or interacts with the software at any time [ISO/IEC 29881:2010 *Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*, 3.5] **4.** person or persons who will ultimately be using the system for its intended purpose [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.13] **5.** individual person who ultimately benefits from the ready-to-use software product functionalities [ISO/IEC 25051:2014 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.7]
cf. direct user, functional user, indirect user, operator, secondary user, user

Note 1 to entry: An end user will generally be defined in terms of a specific software component of a system.

3.1390

endeavor

1. IBD development effort aimed at the delivery of some product or service through the application of a methodology [ISO/IEC 24744:2014 *Software Engineering — Metamodel for development methodologies*, 3.5]

EXAMPLE: projects, programs and infrastructural duties

3.1391

endeavor element

- 1.** simple component of an endeavor [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 3.7]

EXAMPLE: Customer, Invoice (classes), Name, Age (attributes), High-Level Class Model number 17 (a model), System Requirements Description (a document), Coding Cycle number 2, Coding Cycle number 3 (tasks)

Note 1 to entry: During the execution of an endeavor, developers create a number of endeavor elements, such as tasks, models, classes, documents.

3.1392

endurance testing

- 1.** type of performance efficiency testing conducted to evaluate whether a test item can sustain a required load continuously for a specified period of time [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.10]

3.1393

engineering

- 1.** application of a systematic, disciplined, quantifiable approach to structures, machines, products, systems, or processes [*ISO/IEC TR 19759:2016, Software Engineering -- Guide to the Software Engineering Body of Knowledge (SWEBOK)*, 15]

3.1394

engineering change

- 1.** alteration in the configuration of a hardware/software configuration item or items, delivered, to be delivered, or under development, after formal establishment of their configuration identification **2.** in configuration management, an alteration in the configuration of a configuration item or other designated item after formal establishment of its configuration identification

cf. configuration control, engineering change proposal, deviation, waiver

3.1395

engineering change proposal (ECP)

- 1.** in configuration management, a proposed engineering change and the documentation by which the change is described and suggested

cf. configuration control

3.1396

engineering interface reference

- 1.** identifier, in the context of an engineering interface reference management domain, for an engineering object interface that is available for distributed binding [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.16]

Note 1 to entry: An engineering interface reference is necessary to establish distributed bindings, and is distinct from the binding endpoint identifiers used by a basic engineering object for the purposes of interaction.

3.1397

engineering interface reference management domain

- 1.** set of nodes forming a naming domain for the purpose of assigning engineering interface references [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.17]

3.1398

engineering interface reference management policy

- 1.** set of permissions and prohibitions that govern the federation of engineering interface reference management domains [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.18]

3.1399

engineering viewpoint

1. viewpoint on an ODP system and its environment that focuses on the mechanisms and functions required to support distributed interaction between objects in the system [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.1.1.4]

3.1400

enhancement

1. the activities carried out for an application that change the specifications of the application and that also usually change the number of function points as a result [*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*] cf. change

3.1401

enhancement project

1. project to develop and deliver adaptive maintenance [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.22] **2.** a project in which enhancements are made to an existing application [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

Note 1 to entry: In an enhancement project, functionality can be added to, changed in, or deleted from an existing application. An enhancement project can also develop and deliver corrective and perfective maintenance, but these do not contribute to the enhancement project functional size.

3.1402

enhancement project function point count (EFP)

1. count that measures a project that realizes modifications to an existing application [*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.** activity of applying ISO/IEC 20926:2009 to measure the functional size of an enhancement project [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.24]

3.1403

ensure

1. to make certain that things occur or events take place [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2]

cf. assure

Note 1 to entry: Insure is used only for insurance matters.

3.1404

Ent

1. [software] entitlement schema [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.2]

3.1405

Ent creator

entitlement schema creator

1. entity that initially creates an Ent [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.10]

Note 1 to entry: This entity can be part of the organization that created or published the software to which the Ent relates, in which case the Ent creator and software creator will be the same. The Ent creator can also be a separate organization which holds the licensing rights or even a third-party organization unrelated to the software creator (such as in the case where Ents are created for legacy software by a consultant or tool developer).

3.1406

enterprise environmental factors

1. conditions, not under the immediate control of the team, that influence, constrain or direct the project, program or portfolio [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1407

enterprise viewpoint

1. viewpoint on an ODP system and its environment that focuses on the purpose, scope, and policies for that system [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.1.1.1]

3.1408

entitlement schema

Ent

software entitlement schema

1. information structure containing a digital encapsulation of a licensing transaction and its associated entitlement information [ISO/IEC 19770-3:2016, *Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.11]

Note 1 to entry: A single transaction does not necessarily encapsulate a full (or effective) entitlement. An effective entitlement can be determined by an analysis of multiple licensing transactions, of a full license and then of upgrades and/or maintenance transactions assessed together with it.

3.1409

entity

1. a fundamental thing of relevance to the user, about which information is kept [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10] 2. in computer programming, any item that can be named or denoted in a program 3. object (i.e., thing, event or concept) that occurs in a model (i.e., transfer) [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2] 4. object that is to be characterized by measuring its attributes [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.9] 5. representation of a set of real or abstract things that are recognized as the same type because they share the same characteristics and can participate in the same relationships [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.55] 6. concrete or abstract thing of interest [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 6.1] 7. object to be modeled [ISO/IEC 15476-4:2005 *Information technology — CDIF semantic metamodel — Part 4: Data models*, 6.3] 8. logical component of the data store, representing fundamental things of relevance to the user, and about which persistent information is stored [ISO/IEC 29881:2010 *Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*, A.8] 9. registered organization, group within a registered organization, or a project within an organization [ISO/IEC 29110-2-1:2015 — *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.26]

EXAMPLE: a data item, program statement, or subprogram

Note 1 to entry: While in general the word entity can be used to refer to anything, in the context of modeling it is reserved to refer to things in the universe of discourse being modeled.

3.1410

entity component

1. CORBA component with persistent state, identity which is architecturally visible to clients through a primary key, and behavior, which can be transactional [ISO/IEC 19500-3:2012 *Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

3.1411

entity dependent

1. (entity) not meaningful or not significant to the business in and of itself without the presence of other entities, such that an occurrence of entity X must be linked to an occurrence of entity Y, and the deletion of an occurrence of entity Y results in the deletion of all related occurrences of entity X [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.25]

3.1412

entity independent

1. (entity) meaningful or significant to the business in and of itself without the presence of other entities [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.26]

3.1413

entity instance

- 1.** one of a set of real or abstract things represented by an entity [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.56]

Note 1 to entry: Each instance of an entity can be specifically identified by the value of the attribute(s) participating in its primary key [key style]

3.1414

entity-relationship (E-R) diagram

entity-relationship map

- 1.** a diagram that depicts a set of real-world entities and the logical relationships among them
cf. data structure diagram

3.1415

entry (-type)

- 1.** data movement that moves a data group from a functional user across the boundary into the functional process where it is required [ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.8]

Note 1 to entry: an entry is considered to account for certain associated data manipulations (e.g., validation of the entered data)

3.1416

entry criteria

- 1.** states of being that must be present before an effort can begin successfully **2.** artifacts and other review or audit elements that must be completed before the review or audit can be conducted [IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1]
cf. exit criteria

3.1417

entry field

- 1.** area on a screen or in a window in which a user enters data [ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.19]

3.1418

entry point

entrance

entry

- 1.** point in a software module at which execution of the module can begin **2.** point in a test item at which execution of the test item can begin [ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.18]
cf. exit, reentry point

Note 1 to entry: An entry point is an executable statement within a test item that can be selected by an external process as the starting point for one or more paths through the test item. It is most commonly the first executable statement within the test item.

3.1419

entry profile

- 1.** profile targeted at start-up Very Small Entities (i.e., VSEs that started their operation fewer than three years ago) or at VSEs working on small projects (e.g., project size of less than six person-months) [ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.27]

3.1420

enumeration type

- 1.** discrete data type whose members can assume values that are explicitly defined by the programmer
cf. character type, integer type, logical type, real type

EXAMPLE: a data type called COLORS with possible values RED, BLUE, and YELLOW

3.1421

environment

- 1.** [system] context determining the setting and circumstances of all influences upon a system [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.20; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.19; *ISO/IEC/IEEE 42010:2011 Systems and software engineering — Architecture description*, 3.8] **2.** configuration of hardware and software in which the software operates **3.** of an object, the part of the model which is not part of that object [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 8.2] **4.** anything affecting a subject system or affected by a subject system through interactions with it, or anything sharing an interpretation of interactions with a subject system [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description*, 3.6] **5.** concept space, i.e., an area in which a concept has an agreed-to meaning and one or more agreed-to names that are used for the concept [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.57]

Note 1 to entry: The environment of a system includes developmental, technological, business, operational, organizational, political, economic, legal, regulatory, ecological and social influences.

3.1422

environment contract

- 1.** contract between an object and its environment, including Quality of Service constraints, usage and management constraints [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 11.2.3]

3.1423

environmental risk mitigation

- 1.** degree to which a product or system mitigates the potential risk to property or the environment in the intended contexts of use [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.4.3]

3.1424

EO

- 1.** external output [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 4; 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*]

3.1425

epic

- 1.** a high-level or complex user story to be refined into more detailed user stories [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.1426

epilog breakpoint

postamble breakpoint

- 1.** breakpoint that is initiated upon exit from a given program or routine

cf. prolog breakpoint, code breakpoint, data breakpoint, dynamic breakpoint, programmable breakpoint, static breakpoint

3.1427

epoch

- 1.** period of time for which an object displays a particular behavior [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 10.5]

3.1428

EPROM

- 1.** erasable programmable read only memory

3.1429

EQ

- 1.** external inquiry [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 4; 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]

3.1430

equivalence class

1. range on a classification axis which has a rule to judge whether a target system is to be mapped to the range or not [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it, 3.8*]

3.1431

equivalence partition

1. subset of the range of values of a variable, or set of variables, within a test item or at its interfaces, such that all the values in the partition can reasonably be expected to be treated similarly by the test item (i.e., they are considered "equivalent" [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.11*]

3.1432

equivalence partition coverage

1. proportion of identified equivalence partitions of a test item that are covered by a test set [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.12*]

Note 1 to entry: In many cases, the identification of equivalence partitions is subjective (especially in the sub-partitioning of "invalid" partitions), so a definitive count of the number of equivalence partitions in a test item could be impossible.

3.1433

equivalence partitioning

1. test design technique in which test cases are designed to exercise equivalence partitions by using one or more representative members of each partition [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.13*]

3.1434

equivalent faults

1. two or more faults that result in the same failure mode

3.1435

equivalent IDL

1. client mappings; that is, mappings of the externally-visible component features for component declarations, or home features for home declarations [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

Note 1 to entry: Implicitly defined by a component definition in IDL (interface definition language)

3.1436

equivalent interface

1. interface that manifests the component's or home's surface features to clients, allowing clients to navigate among the component's facets, and to connect to the component's ports, as defined by the component's or home's equivalent interface definition language [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.1437

ERA

1. Entity-Relationship-Attribute modeling [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 5.2*]

3.1438

erasable programming read only memory (EPROM)

1. type of programmable ROM which can be rewritten after erasing the existing data using ultraviolet (UV) rays

Note 1 to entry: The device can be rewritten many times.

3.1439

ergonomics

1. scientific discipline concerned with the understanding of the interactions among human and other elements of a system. **2.** profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance.

3.1440

errata

1. severe service-disrupting bugs for which there is no known workaround

Note 1 to entry: Fixes for such bugs can often be introduced on a frozen branch.

3.1441

error

1. human action that produces an incorrect result [*IEEE 1044-2009 IEEE Standard Classification for Software Anomalies, 2*] **2.** difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition **3.** erroneous state of the system [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.4.4*] cf. failure, defect

EXAMPLE: omission or misinterpretation of user requirements in a software specification, incorrect translation, or omission of a requirement in the design specification

3.1442

error guessing

1. test design technique in which test cases are derived on the basis of the tester's knowledge of past failures, or general knowledge of failure modes [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.14*]

3.1443

error message

1. a message that the application gives when incorrect data is entered or when another processing error occurs [*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*]

3.1444

error model

error prediction model

1. in software evaluation, a model used to estimate or predict the number of remaining faults, required test time, and similar characteristics of a system

3.1445

error prediction

1. quantitative statement about the expected number or nature of faults in a system or component cf. error model, error seeding

3.1446

error processing

1. process of detecting and responding to a program's errors

3.1447

error seeding

bug seeding

fault seeding

1. process of intentionally adding known faults to those already in a computer program for the purpose of monitoring the rate of detection and removal, and estimating the number of faults remaining in the program cf. indigenous error

3.1448

error tolerance

- 1.** ability of a system or component to continue normal operation despite the presence of erroneous inputs
cf. fault tolerance, robustness

3.1449

ES

- 1.** early start date [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1450

escaped

- 1.** preceding each occurrence of a pattern by the <EscapeCharacter>, if it is necessary to include a pattern in the text string that matches the <CloseText> delimiter [*ISO/IEC 15475-3:2002 Information technology — CDIF transfer format — Part 3: Encoding ENCODING.1, 7.2.11*]

3.1451

escrow

- 1.** source code and documentation that is kept in the custody of a third party until specified contractual conditions have been fulfilled [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.20*]

3.1452

ESIOP

- 1.** environment specific inter-ORB protocol [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 5*]

3.1453

ESOH

- 1.** environment, safety, and occupational health [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.1454

establish and maintain

- 1.** to formulate, document, and use [a policy or procedure] throughout an organization
cf. maintain

Note 1 to entry: This phrase means more than a combination of its component terms; it includes documentation and usage.

3.1455

established requirement

- 1.** requirement that the project has verified as satisfying project-specific criteria (such as clarity, suitability, and feasibility) and has validated to be an accurate representation of stakeholder needs, wants, and expectations [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*]

Note 1 to entry: Established requirements are accepted by the project to form the basis of product development.

3.1456

establishing behavior

- 1.** behavior by which a given contract is put in place between given objects [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.2.1*]

3.1457

estimate

- 1.** a quantitative assessment of the likely amount or outcome. Usually applied to project costs, resources, effort, and durations and is usually preceded by a modifier (i.e., preliminary, conceptual, feasibility, order-of-magnitude, definitive). It should always include some indication of accuracy (e. g., (+ or -) x percent) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. budget, cost

3.1458

estimate activity durations

1. the process of approximating the number of work periods needed to complete individual activities with estimated resources [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1459

estimate activity resources

1. the process of estimating the type and quantities of material, people, equipment or supplies required to perform each activity [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1460

estimate at completion (EAC)

1. the expected total cost of completing all work expressed as the sum of the actual cost to date and the estimate to complete [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. earned value technique, estimate to complete

3.1461

estimate costs

1. the process of developing an approximation of the monetary resources needed to complete project activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1462

estimate to complete (ETC)

1. the expected cost to finish all the remaining project work [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. estimate at completion

3.1463

estimated function point count

1. possible function point count in an early phase of an application's life cycle to determine the size of an application or a project in which certain minimum specifications are assumed [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

Note 1 to entry: Typically, the number of functions is recorded per type, and a default value is used for the complexity average for the transactional functions (transactions) and low for the data functions (logical files).

3.1464

ETC

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

3.1465

EV

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

3.1466

evaluation

1. systematic determination of the extent to which an entity meets its specified criteria [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.12; ISO/IEC 25001:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Planning and management, 4.1*] 2. action that assesses the value of something [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.6.7*]

EXAMPLE: the action by which an ODP system assigns a relative status to something, according to estimation by the system

Note 1 to entry: Value can be considered in terms of usefulness, importance, preference, acceptability, etc.; the evaluated target can be, for example, a credit rating, a system state, a potential behavior.

3.1467

evaluation activity

- 1.** assessment of systems or software product against targeted values of identified and applicable quality characteristics performed using applicable techniques or methods [*ISO/IEC 25001:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Planning and management, 4.1*]

3.1468

evaluation checklist

- 1.** list of questions, each of which is designed to check for conformity of a product, process or service to one or more provisions within a particular International Standard [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.2*]

3.1469

evaluation coverage

- 1.** degree to which the evaluation covers the specified software product quality requirements [*ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.17*]

3.1470

evaluation group

- 1.** organization responsible for specifying the systems and software quality requirements as well as managing and implementing the quality evaluation activities through the provision of technology, tools, experiences, and management skills [*ISO/IEC 25001:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Planning and management, 4.3*]

Note 1 to entry: Software quality requirements could be specified previously by the requestor of the evaluation, while the evaluation group would verify presence and value of the software quality requirements.

3.1471

evaluation level

- 1.** rigor to be applied during the evaluation that defines the depth or thoroughness of the evaluation in terms of evaluation techniques to be applied and evaluation results to be achieved [*ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.18*]

3.1472

evaluation method

- 1.** procedure describing actions to be performed by the evaluator in order to obtain results for the specified measurement applied to the specified product components or on the product as a whole [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.8; ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.20*]

3.1473

evaluation module

- 1.** package of evaluation technology for measuring software quality characteristics, subcharacteristics, or attributes [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.9; ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.20*]

Note 1 to entry: The package includes evaluation methods and techniques, input to be evaluated, data to be measured and collected, and supporting procedures and tools.

3.1474

evaluation module (EVM)

- 1.** microcomputer module used in application development, e.g., to benchmark software, prototype applications, and debug algorithms for computer systems

3.1475

evaluation procedure

1. series of tasks and steps that, when completed, enable the evaluation team to determine if the product, process or service being evaluated is conformant to a particular standard [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.3*]

3.1476

evaluation records

1. documented objective evidence of all activities performed and of all results achieved within the evaluation process [*ISO/IEC 14598-5:1998 Information technology — Software product evaluation — Part 5: Process for evaluators, 4.3; ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.21*]

3.1477

evaluation report

1. system follow-up report that describes how the system objectives have been met, identifies the remaining problems, and is intended to assist future development [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** document that presents evaluation results and other information relevant to an evaluation [*ISO/IEC 14598-5:1998 Information technology — Software product evaluation — Part 5: Process for evaluators, 4.2*]

3.1478

evaluation requester

1. person or organization that requests an evaluation [*ISO/IEC 14598-5:1998 Information technology — Software product evaluation — Part 5: Process for evaluators, 4.4; ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.22*]

3.1479

evaluation sponsor

1. person or organization that requires the evaluation to be performed and provides financial or other resources to carry it out [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.4*]

3.1480

evaluation stringency

1. degree required for the software product quality characteristics and subcharacteristics to fulfill the expected use criticality of the software product [*ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.24*]

3.1481

evaluation technology

technology used for evaluation

1. techniques, processes, tools, measures and relevant technical information used for evaluation [*ISO/IEC 25001:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Planning and management, 4.4*]

EXAMPLE: internal, external or quality in use measures or specific evaluation processes designed for developers, acquirers or independent evaluators

3.1482

evaluation tool

1. instrument that can be used during evaluation to collect data, to perform interpretation of data or to automate part of the evaluation [*ISO/IEC 14598-5:1998 Information technology — Software product evaluation — Part 5: Process for evaluators, 4.5*]

EXAMPLE: source code analyzers to compute code metrics, CASE tools to produce formalized models, test environments to run the executable programs, checklists to collect inspection data, or spreadsheets to produce syntheses of measures

3.1483

evaluator

- 1.** individual or organization that performs an evaluation [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.10*] **2.** organization that performs an evaluation [*ISO/IEC 14598-5:1998 Information technology — Software product evaluation — Part 5: Process for evaluators, 4.6*]

EXAMPLE: a testing laboratory, the quality department of a software development organization, a government organization, or a user

3.1484

event

- 1.** occurrence of a particular set of circumstances [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.2*] **2.** external or internal stimulus used for synchronization purposes **3.** change detectable by the subject software [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*] **4.** fact that an action has taken place [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.4*] **5.** singular moment in time at which some perceptible phenomenological change (energy, matter, or information) occurs at the port of a unit [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.12*]

Note 1 to entry: The event can be certain or uncertain. The event can be a single occurrence or a series of occurrences. The probability associated with the event can be estimated for a given period of time. An event can be an external interrupt, a timer expiration, an internal signal, or an internal message.

[SOURCE: ISO Guide 73:2009, definition 3.6.1.3]

3.1485

event history

- 1.** object representing significant actions [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 13.1.1.1*]

3.1486

event sequence analysis

- 1.** per

3.1487

event sequence diagram

- 1.** diagram that identifies the sequence of tasks required to process an external event

3.1488

event sink

- 1.** operation interface consuming announcements carrying notifications of typed events [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.16*] **2.** named connection point into which events of a specified type can be pushed [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]
cf. consumer

3.1489

event source

- 1.** operation interface originating announcements carrying notifications of typed events [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.16*] **2.** named connection point that emits events of a specified type to one or more interested event consumers, or to an event channel [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.1490

event synchronization

- 1.** control of task activation by means of signals

Note 1 to entry: Three types of event synchronization are possible: external interrupts, timer expiration, and internal signals from other tasks.

3.1491

event trace

- 1.** time-ordered description of each external input and the time at which it occurred

3.1492

event-sequencing logic

- 1.** description of how a task responds to each of its message or event inputs

Note 1 to entry: in particular, what output is generated as a result of each input.

3.1493

EventItem

- 1.** occurrence of an EventType [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1494

EventType

- 1.** set of possible times or time periods at which an Event can occur [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1495

EVM

- 1.** earned value management [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*] **2.** evaluation module

3.1496

exception

- 1.** event that causes suspension of normal program execution **2.** indication that an operation request was not performed successfully [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces, 5.3.8*]

Note 1 to entry: Types include addressing exception, data exception, operation exception, overflow exception, protection exception, and underflow exception.

3.1497

exception handling

- 1.** programming language mechanism that passes error information by throwing and catching exceptions

3.1498

exclusive requirement

- 1.** requirement of a normative document that must necessarily be fulfilled in order to comply with that document [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.5*]

3.1499

executable requirements specification

- 1.** software requirement specification that is represented in an executable requirements language

3.1500

executable source statement

- 1.** source statement that directs the actions of the computer at run time

3.1501

executable statement

- 1.** statement which, when compiled, is translated into object code, which will be executed procedurally when the test item is running and can perform an action on program data [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.19*]

3.1502

execute

1. to carry out an instruction, process, or computer program **2.** directing, managing, performing, and accomplishing the project work, providing the deliverables, and providing work performance information [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1503

executing process group

1. those processes performed to complete the work defined in the project management plan to satisfy the project specifications [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1504

execution efficiency

1. degree to which a system or component performs its designated functions with minimum consumption of time
cf. execution time, storage efficiency

3.1505

execution time

1. time which elapses between task submission and completion [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.7*] **2.** amount of elapsed time or processor time used in executing a computer program
cf. run time

Note 1 to entry: Processor time is usually less than elapsed time because the processor can be idle (for example, awaiting needed computer resources) or employed on other tasks during the execution of a program.

3.1506

execution trace

code trace

control flow trace

1. record of the sequence of instructions executed during the execution of a computer program
cf. retrospective trace, subroutine trace, symbolic trace, variable trace

Note 1 to entry: often takes the form of a list of code labels encountered as the program executes

3.1507

executor

1. programming artifacts that supply the behavior of a component or a component home [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.1508

existence constraint

1. constraint stating that an instance of one entity cannot exist unless an instance of another related entity also exists [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.59*]

Note 1 to entry: [key style]

3.1509

existence dependency

1. constraint between two related entities indicating that no instance of one can exist without being related to an instance of the other [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.60*]

Note 1 to entry: The following association types represent existence dependencies: identifying relationships, categorization structures and mandatory nonidentifying relationships. [key style]

3.1510

existing software

1. software that is already developed and available; is usable either "as is" or with modifications; and which is provided by the supplier, acquirer, or a third party

3.1511

Exit

exit type

- 1.** point in a software module at which execution of the module can terminate
2. data movement that moves a data group from a functional process across the boundary to the functional user that requires it [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.9*]
cf. entry point, return

Note 1 to entry: An exit is considered to account for certain associated data manipulations (e.g. formatting and routing associated with the data to be exited).

3.1512

exit criteria

- 1.** states of being that must be present before an effort can end successfully
2. review or audit elements that must be assessed, completed, and action items closed before successful completion of the technical review or audit can be declared [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1*]
cf. entry criteria

3.1513

exit point

- 1.** last executable statement within a test item [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.20*]

Note 1 to entry: An exit point is a terminal point of a path through a test item, being an executable statement within the test item which either terminates the test item, or returns control to an external process. This is most commonly the last executable statement within the test item.

3.1514

exit routine

- 1.** routine that receives control when a specified event, such as an error, occurs

3.1515

expandability

- 1.** degree of effort required to improve or modify software functions' efficiency
cf. extendability

3.1516

expected monetary value (EMV) Analysis

- 1.** a statistical technique that calculates the average outcome when the future includes scenarios that may or may not happen. A common use of this technique is within decision tree analysis. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1517

expected results

- 1.** observable predicted behavior of the test item under specified conditions based on its specification or another source [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.15*]

3.1518

expected value

50-50 estimate

- 1.** estimated outcome that is as likely to be exceeded as not

Note 1 to entry: the mean of the probability distribution, the point where the cumulative probability function equals 0.5

3.1519

expected value of perfect information

- 1.** in decision-tree analysis, the difference between the expected value of the decision tree and the value of the decision tree if all random outcomes were known in advance

Note 1 to entry: helps the decision maker determine whether it is justifiable to invest in activities that would reduce uncertainties

3.1520

expert judgment

- 1.** judgment provided based upon expertise in an application area, knowledge area, discipline, industry, etc. as appropriate for the activity being performed. Such expertise may be provided by any group or person with specialized education, knowledge, skill, experience, or training [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1521

expert system (ES)

- 1.** computer system that provides for expertly solving problems in a given field or application area by drawing inferences from a knowledge base developed from human expertise [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: Some expert systems are able to improve their knowledge base and develop new inference rules based on their experience with previous problems.

3.1522

explanatory report

- 1.** document attached to a product for providing complementary information in order to assist understanding and to avoid inappropriate usage of the product [*ISO/IEC 29155-3:2015 Systems and software engineering — Information technology project performance benchmarking framework — Part 3: Guidance for reporting*]

Note 1 to entry: Examples of an explanatory report are data element definitions, data demographics, data source information which are attached to benchmarking repositories or benchmarks. Examples of the product are benchmarking repository, benchmark(s), or software tools to support benchmarking activities.

3.1523

exploratory testing

- 1.** type of unscripted experience-based testing in which the tester spontaneously designs and executes tests based on the tester's existing relevant knowledge, prior exploration of the test item (including the results of previous tests), and heuristic "rules of thumb" regarding common software behaviors and types of failure [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes, 4.9*]

Note 1 to entry: Exploratory testing hunts for hidden properties (including hidden behaviors) that, while quite possibly benign by themselves, could interfere with other properties of the software under test, and so constitute a risk that the software will fail.

3.1524

export process

- 1.** process of generating a transfer file from a source environment [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.1*]

3.1525

exporter

- 1.** agent of the export process [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.1*]

3.1526

extend

- 1.** in UML, a relationship from an extending use case to a base use case, specifying how the behavior defined for the extending use case can be optionally inserted into the behavior defined for the base use case

3.1527

extendability

extensibility

- 1.** ease with which a system or component can be modified to increase its storage or functional capacity
cf.: expandability, flexibility, maintainability

3.1528

extended component

- 1.** component that offers any type of port [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.1529

extended element

- 1.** element within a tag that provides additional information beyond that documented explicitly in the standard

3.1530

extended entry table

- 1.** decision table where the conditions and actions are generally described but are incomplete [*ISO 5806:1984 Information processing — Specification of single-hit decision tables*, 3.15]

Note 1 to entry: The specifications are completed by the values specified in the rules

3.1531

extended process set

- 1.** set of processes specific to a maturity level higher than the basic maturity level that ensures the achievement of the relevant process profile [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.3.5]

3.1532

extensible markup language (XML)

- 1.** license-free and platform-independent markup language that carries rules for generating text formats that contain structured data [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.15]

3.1533

extension of a type

- 1.** set of entities that satisfy the type at any particular time [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.3.4]

3.1534

extensional set

current extent

- 1.** set containing the currently existing instances of a class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.61]

Note 1 to entry: The instances in the extensional set correspond to the database and data modeling notion of instance.

3.1535

external

- 1.** input information source or output information destination that is outside the scope of the project life cycle.
cf. invocation, iteration, mapping

3.1536

external attribute

- 1.** measurable property of an entity which can only be derived with respect to how it relates to its environment

Note 1 to entry: External attributes are those that relate to requirements (external properties of the software). External attributes can only be derived from the operational behavior of the system of which it is a part.

3.1537

external dependency

- 1.** a relationship between project activities and non-project activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1538

external event

- 1.** event from an external object, typically an interrupt from an external I/O device

3.1539

external I/O device

- 1.** hardware input and/or output device that is outside the software system and part of the external environment

3.1540

external input (EI)

- 1.** a unique function recognized by the user in which data and/or control information from outside the application is entered into the application [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*] **2.** elementary process that processes data or control information sent from outside the boundary [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.27*]
cf. external inquiry, external output

Note 1 to entry: The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system. An external input is a type of base functional component.

3.1541

external inquiry (EQ)

- 1.** a unique input/output combination recognized by the user in which the application distributes an output fully determined in size without further data processing, as a result of the input [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*] **2.** elementary process that sends data or control information outside the boundary [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.28*]
cf. external input, external output

Note 1 to entry: The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF or EIF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered. An external inquiry is a type of base functional component.

3.1542

external interface file (EIF)

- 1.** a logical group of permanent data seen from the perspective of the user that an application uses but that a different application maintains [*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.** user-recognizable group of logically related data or control information, which is referenced by the application being measured, but which is maintained within the boundary of another application [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.29*]
cf. internal logical file

Note 1 to entry: The primary intent of an EIF is to hold data referenced through one or more elementary processes within the boundary of the application counted. This means an EIF counted for an application must be in an ILF in another application. An external interface file is a type of base functional component.

3.1543

external interface requirement

- 1.** system or software requirement that specifies a hardware, software, or database element with which a system/software system or system/software component must interface, or that sets forth constraints on formats, timing, or other factors caused by such an interface

3.1544

external measure

- 1.** indirect measure of a product derived from measures of the behavior of the system of which it is a part

Note 1 to entry: The number of failures found during testing is an external measure of the number of faults in the program, because the number of failures is counted during the operation of a computer system running the program. External measures can be used to evaluate quality attributes closer to the ultimate objectives of the design.

3.1545

external measure of software quality

- 1.** measure of the degree to which a software product enables the behavior of a system under specified conditions to satisfy stated and implied needs for the system, including the software to be used under specified conditions [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.3.5*]

cf. external software quality, internal measure of software quality

EXAMPLE: The number of failures found during testing is an external measure of software quality related to the internal measure, the number of faults present in the computer system. The two measures are not necessarily identical since testing might not find all faults, and a fault can give rise to apparently different failures in different circumstances.

Note 1 to entry: Attributes of the behavior can be verified or validated by executing the software product during testing and operation.

3.1546

external measure of system or software quality

1. measure of the degree to which a system or software product enables the behavior to satisfy stated and implied needs for the system, including the software to be used under specified conditions [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.11]

EXAMPLE: The number of failures found during testing is an external measure of software quality related to the number of faults present in the computer system. The two measures are not necessarily identical since testing does not find all faults, and a fault can give rise to apparently different failures in different circumstances.

Note 1 to entry: Attributes of the behavior can be verified or validated by executing the system or software product during testing and operation.

3.1547

external output (EO)

1. a unique output recognized by the user which crosses the application boundary [*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.** elementary process that sends data or control information outside the application's boundary and includes additional processing logic beyond that of an external inquiry [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.30]

cf. external input, external inquiry

Note 1 to entry: The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information. The processing logic must contain at least one mathematical formula or calculation, or create derived data. An external output can also maintain one or more ILFs and/or alter the behavior of the system. An external output is a type of base functional component.

3.1548

external quality

1. extent to which a product satisfies stated and implied needs when used under specified conditions

3.1549

external variability

1. variability that is visible to customers [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.4]

3.1550

extranet

1. set of intranets connected for specific objectives, spanning multiple organizations

3.1551

F-profile

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

3.1552

facet

1. operation interface in which a computational component plays a server role [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 7.1.16] **2.** distinct named interface

provided by the component for client interaction [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

Note 1 to entry: the primary vehicle through which a component exposes its functional application behavior to clients during normal execution

3.1553

faceted search

1. progressive search which allows users to narrow the results by selecting values for one or more attributes [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.8]

3.1554

facilitated workshops

1. an elicitation technique using focused sessions that bring key cross-functional stakeholders together to define product requirements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1555

facility

1. physical means or equipment for facilitating the performance of an action [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.13; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.21; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.20]

EXAMPLE: buildings, instruments, tools

3.1556

factoring

1. process of decomposing a system into a hierarchy of modules **2.** process of removing a function from a module and placing it into a module of its own
cf. modular decomposition

3.1557

factory

1. object that, in response to an interaction initiated by its environment, creates a new object and returns a reference to it to the environment [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.25]

3.1558

fail safe

1. pertaining to a system or component that automatically places itself in a safe operating mode in the event of a failure
cf. fail soft, fault secure, fault tolerance

EXAMPLE: a traffic light that reverts to blinking red in all directions when normal operation fails

3.1559

fail soft

1. pertaining to a system or component that continues to provide partial operational capability in the event of certain failures
cf. fail safe, fault secure, fault tolerance

EXAMPLE: a traffic light that continues to alternate between red and green if the yellow light fails

3.1560

failure

1. termination of the ability of a system to perform a required function or its inability to perform within previously specified limits; an externally visible deviation from the system's specification [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.4.8] **2.** violation of a contract [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.6.1]

Note 1 to entry: A failure can be produced when a fault is encountered.

3.1561

failure mode

- 1.** physical or functional manifestation of a failure

EXAMPLE: a system in failure mode is characterized by slow operation, incorrect outputs, or complete termination of execution

3.1562

failure mode and effect analysis (FMEA)

- 1.** an analytical procedure in which each potential failure mode in every component of a product is analyzed to determine its effect on the reliability of that component and, by itself or in combination with other possible failure modes, on the reliability of the product or system and on the required function of the component; or the examination of a product (at the system and/or lower levels) for all ways that a failure may occur. For each potential failure, an estimate is made of its effect on the total system and of its impact. In addition, a review is undertaken of the action planned to minimize the probability of failure and to minimize its effects [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1563

failure rate

failure ratio

- 1.** ratio of the number of failures of a given category to a given unit of measure

EXAMPLE: failures per unit of time, failures per number of transactions, failures per number of computer runs

3.1564

failure transparency

- 1.** distribution transparency which masks, from an object, the failure and possible recovery of other objects (or itself), to enable fault tolerance [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 4.4.1.2*]

3.1565

fallback plan

- 1.** an alternative set of actions and tasks available in the event the primary plan must be abandoned because of issues, risks or other causes [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1566

families of programs

- 1.** sets of programs that are related by sharing significant portions of requirements, design, and code

Note 1 to entry: a program family might include one version of a program developed for an English-speaking audience, a second version of a program developed for a German-speaking audience, and a third version for a Japanese-speaking audience

3.1567

fast tracking

- 1.** a schedule compression technique in which activities or phases normally done in sequence are performed in parallel for at least a portion of their duration [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. schedule compression, crashing

3.1568

fatal error

- 1.** error that results in the complete inability of a system or component to function

3.1569

fault

bug

- 1.** manifestation of an error in software **2.** incorrect step, process, or data definition in a computer program [*ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and*

Evaluation (SQuaRE) — Evaluation process, 4.27] 3. situation that can cause errors to occur in an object [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.6.3] 4. defect in a hardware device or component 5. defect in a system or a representation of a system that if executed/activated could potentially result in an error [ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.4.5]

Note 1 to entry: A fault, if encountered, can cause a failure. Faults can occur in specifications when they are not correct.

3.1570

fault dictionary

- 1.** a list of faults in a system or component, and the tests that have been designed to detect them

3.1571

fault isolation

- 1.** ability of a subsystem to prevent a fault within the subsystem from causing consequential faults in other subsystems

3.1572

fault masking

- 1.** condition in which one fault prevents the detection of another

3.1573

fault secure

- 1.** pertaining to a system or component in which no failures are produced from a prescribed set of faults
cf. fault tolerance, fail-safe, fail soft

3.1574

fault tolerance

- 1.** degree to which a system, product or component operates as intended despite the presence of hardware or software faults [ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.5.3] **2.** pertaining to the study of errors, faults, and failures, and of methods for enabling systems to continue normal operation in the presence of faults

cf. error tolerance, fail safe, fail soft, fault secure, robustness

3.1575

fault-tolerant

- 1.** pertaining to a system or component that is able to continue normal operation despite the presence of faults

3.1576

FCA

- 1.** functional configuration audit

3.1577

FD

- 1.** full deployment [IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2]

3.1578

FDC

- 1.** functional domain categorization [ISO/IEC TR 14143-5:2004 Information technology — Software measurement — Functional size measurement — Part 5: Determination of functional domains for use with functional size measurement, 4]

3.1579

FDT

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

3.1580

feasibility

1. degree to which the requirements, design, or plans for a system or component can be implemented under existing constraints

3.1581

feasibility study

1. study to identify and analyze a problem and its potential solutions in order to determine their viability, costs, and benefits. [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.1582

feature

1. distinguishing characteristic of a system item **2.** functional or non-functional distinguishing characteristic of a system, often an enhancement to an existing system [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.9; ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.6] **3.** abstract functional characteristic of a system of interest that end-users and other stakeholders can understand [ISO/IEC 26550:2015 *Software and systems engineering — Reference model for product line engineering and management*, 3.14]

3.1583

feature branch

1. branch created for developing a particular set of features

Note 1 to entry: The branch is typically not released but is collapsed back at some point to its parent branch.

3.1584

feature freeze

1. period during which no new features are added to a specific branch

Note 1 to entry: allows the branch to stabilize for a release.

3.1585

feature reference

1. expression that unambiguously identifies a diagram feature in a diagram [IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.48]

3.1586

feature set

1. collection of items which contain the test conditions of the test item to be tested which can be collected from risks, requirements, functions, models, etc. [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 1] **2.** logical subset of the test item(s) that could be treated independently of other feature sets in the subsequent test design activities [ISO/IEC/IEEE 29119-2:2013 *Software and systems engineering — Software testing — Part 2: Test processes*, 4.10]

3.1587

fee

1. profit as a component of compensation to a seller [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.1588

fetch

1. to locate and load computer instructions or data from storage
cf. move, store

3.1589

FF

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

3.1590

FFP

- 1.** firm fixed price (contract) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1591

field of application (of a specification)

- 1.** properties the environment of the ODP system must have for the specification of that system to be used [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language*, 6.1.2]

3.1592

field programmable gate array (FPGA)

- 1.** logic device designed to be programmed after it is acquired

Note 1 to entry: often based on look-up table architecture

3.1593

fieldbus

- 1.** industrial computer network protocol used for real-time distributed control

Note 1 to entry: a family of related standardized interfaces

3.1594

fifth-generation language (5GL)

- 1.** computer language that incorporates the concepts of knowledge-based systems, expert systems, inference engines, and natural language processing

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

3.1595

figurative constant

- 1.** data name that is reserved for a specific constant in a programming language

cf. literal

EXAMPLE: The data name THREE is reserved to represent the value 3.

3.1596

file

- 1.** set of related records treated as a unit **2.** named set of records stored or processed as a unit [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: In stock control, a file could consist of a set of invoice records.

3.1597

file type referenced (FTR)

- 1.** data function read or maintained by a transactional function [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.31*] **2.** an internal logical file (ILF) or an external interface file (EIF) maintained or read by a transaction [*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*]

3.1598

final function point count

- 1.** a count to determine the number of function points at the end of a project [*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*]

3.1599

final transfer set

1. collection of changed objects that are to be transferred integrally to one or more production environments, including implementation instructions [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.18*]

3.1600

financial independence

1. of software quality assurance (SQA), situation in which control of the SQA budget is vested in an organization independent of the development organization [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*]

3.1601

finish date

1. a point in time associated with a schedule activity's completion. Usually qualified by one of the following: actual, planned, estimated, scheduled, early, late, baseline, target, or current [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1602

finish-to-finish (FF)

1. a logical relationship in which a successor activity cannot finish until a predecessor activity has finished [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] cf. logical relationship

3.1603

finish-to-start (FS)

1. a logical relationship in which a successor activity cannot start until a predecessor activity has finished [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] cf. logical relationship

3.1604

finite state machine

1. computational model consisting of a finite number of states and transitions between those states, possibly with accompanying actions

3.1605

firm-fixed-price contract (FFP)

1. a type of fixed price contract where the buyer pays the seller a set amount (as defined by the contract), regardless of the seller's costs [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1606

firmware

1. combination of a hardware device and computer instructions or computer data that reside as read-only software on the hardware device [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.14*] **2.** ordered set of instructions and associated data stored in a way that is functionally independent of main storage, usually in a ROM [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: The software cannot be readily modified under program control.

3.1607

first input routine

1. those activities required to obtain the logical record, if any, to be processed first

3.1608

FiSMA

1. Finnish Software Measurement Association [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, A.10*]

Note 1 to entry: a network of Finnish companies, which share interest in developing software measurement and/or software processes.

3.1609

fixed cost

1. cost that is not dependent on the rate of production

cf. variable cost

Note 1 to entry: such as facilities cost or loan interest

3.1610

fixed formula method

1. an earned value method for assigning a specified percentage of budget value of a work package to the start milestone of the work package with the remaining budget value percentage assigned when the work package is complete [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1611

fixed price with economic price adjustment contract (FP-EPA)

1. A fixed-price contract, but with a special provision allowing for pre-defined final adjustments to the contract price due to changed conditions, such as inflation changes, or cost increases (or decreases) for specific commodities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1612

fixed-cost analysis

1. analysis that seeks to maximize the effectiveness that can be attained from a fixed, maximum investment

cf. fixed-effectiveness analysis

3.1613

fixed-effectiveness analysis

1. analysis that seeks to minimize the investment needed to attain a fixed, minimum degree of effectiveness

cf. fixed-cost analysis

3.1614

fixed-price contract

1. an agreement that sets the amount that will be paid for a defined scope of work regardless of cost or effort to deliver it [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1615

fixed-price-incentive-fee (FPIF) contract

1. a type of contract where the buyer pays the seller a set amount (as defined by the contract), and the seller can earn an additional amount if the seller meets defined performance criteria [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1616

flag

1. variable that is set to a prescribed state, often 'true' or 'false,' based on the results of a process or the occurrence of a specified condition

cf. indicator, semaphore

3.1617

flash memory

NVRAM

non-volatile random access memory

1. larger and faster programmable ROM which allows data to be electrically erased from memory and rewritten many times

cf. EEPROM

flexibility

1. ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed **2.** degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in contexts beyond those initially specified in the requirements [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.1.5.2*]

cf. adaptability, extendability, maintainability

Note 1 to entry: Flexibility enables products to take account of circumstances, opportunities and individual preferences that had not been anticipated in advance. If a product is not designed for flexibility, it might not be safe to use the product in unintended contexts. Flexibility can be measured either as the extent to which a product can be used by additional types of users to achieve additional types of goals with effectiveness, efficiency, freedom from risk and satisfaction in additional types of contexts of use, or by a capability to be modified to support adaptation for new types of users, tasks and environments, and suitability for individualization.

3.1618

flip-flop

latch

1. electronic circuit with one or two stable states

Note 1 to entry: can be used to store 0 or 1 as digital data

3.1619

float

slack

1. amount of unscheduled time between sequential activities not on the critical path, which can be used to delay the completion of the earlier activity or advance the start of the later activity

cf. free float, total float

3.1620

flow

1. abstraction of a sequence of interactions, resulting in conveyance of information from a producer object to a consumer object [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.*]

EXAMPLE: multimedia data broadcast

Note 1 to entry: A flow can be used to abstract over, for example, the exact structure of a sequence of interactions, or over a continuous interaction including the special case of an analogue information flow.

3.1621

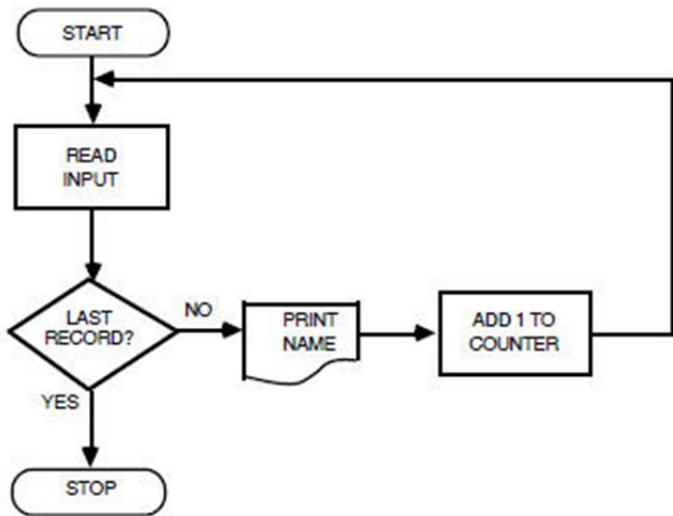
flowchart

flow diagram

1. graphical representation of a process or the step-by-step solution of a problem, using suitably annotated geometric figures connected by flowlines for the purpose of designing or documenting a process or program [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** graphical representation of the definition, analysis, or method of solution of a problem in which symbols are used to represent operations, data, flow, equipment, etc.

[*ISO 5807:1985 Information processing — Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts, 3.3*] **3.** control flow diagram in which suitably annotated geometrical figures are used to represent operations, data, or equipment, and arrows are used to indicate the sequential flow from one to another **4.** the depiction in a diagram format of the inputs, process actions, and outputs of one or more processes within a system [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. block diagram, box diagram, bubble chart, graph, input-process-output chart, structure chart

**Figure 13 — Flowchart**

3.1622 flowcharter

1. software tool that accepts as input a design or code representation of a program and produces as output a flowchart of the program

3.1623 FMEA

1. failure mode and effect analysis [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1624 FMECA

1. failure mode, effects, and criticality analysis [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.1625 FOC

1. full operational capability [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.1626 focus groups

1. an elicitation technique that brings together prequalified stakeholders and subject matter experts to learn about their expectations and attitudes about a proposed product, service, or result [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1627 footer

1. material repeated at the bottom of each page

EXAMPLE: section title

3.1628

For Exposition Only (FEO) page

1. model page that contains pictorial and graphical information (in contrast to text) about a specific diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.51*]

Note 1 to entry: Unlike a diagram, the contents of a For Exposition Only page (FEO page) need not comply with IDEF0 rules.

3.1629

forecast

1. estimate or prediction of conditions and events in the project's future based on information and knowledge available at the time of the forecast. The information is based on the project's past performance and expected future performance, and includes information that could impact the project in the future, such as estimate at completion and estimate to complete [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1630

foreground

1. in job scheduling, the computing environment in which high-priority processes or those requiring user interaction are executed

cf. background, foreground processing

3.1631

foreground processing

1. execution of a high-priority process while lower priority processes await the availability of computer resources, or the execution of processes that require user interaction

cf. background processing

3.1632

foreign key

migrated key

1. attribute, or combination of attributes, of a child or category entity instance whose values match those in the primary key of a related parent or generic entity instance [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.62]

Note 1 to entry: A foreign key results from the migration of the parent or generic entity's primary key through a generalization structure or a relationship. [key style]

3.1633

forking action

1. dividing action, where the enabled chains must (subject to failure) eventually join each other, i.e., the enabled chains cannot join other chains and they cannot terminate separately. [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.1.5]

3.1634

form

1. module or formulary to collect data [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.20]

Note 1 to entry: It can be paper-based (paper form) or digital.

3.1635

form, fit, and function

1. in configuration management, that configuration comprising the physical and functional characteristics of an item as an entity, but not including any characteristics of the elements making up the item

cf. configuration identification

3.1636

formal design

1. process of using a formal method for software design

3.1637

formal evaluation process

1. structured approach to evaluating alternative solutions against established criteria to determine a recommended solution to address an issue

3.1638

formal language

artificial language

1. language whose rules are explicitly established prior to its use [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

cf. natural language

EXAMPLE: programming languages and mathematical languages

3.1639

formal parameter

1. variable used in a software module to represent data or program elements that are to be passed to the module by a calling module

cf. argument (3)

3.1640

formal qualification review (FQR)

1. test, inspection, or analytical process by which a group of configuration items comprising a system is verified to have met specific contractual performance requirements

cf. code review, design review, requirements review, test readiness review

3.1641

formal requirements language

verifiable requirements language

1. artificial language used to represent a software requirement

Note 1 to entry: The resulting formal requirements can be proven "correct" through proof-of-correctness methods.

3.1642

formal specification

1. specification that is used to prove mathematically the validity of an implementation or to derive mathematically the implementation [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **2.** specification written in a formal notation, often for use in proof of correctness **3.** specification written and approved in accordance with established standards

3.1643

formal testing

1. testing conducted in accordance with test plans and procedures that have been reviewed and approved by a customer, user, or designated level of management

cf. informal testing

3.1644

formalization

1. precise description of the semantics of a language in terms of a formal language such as first order logic [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.63]

3.1645

formative construct

1. construct that is formed from its observed measures in the relationship between a construct and its measures [ISO/IEC 33003:2015 *Information technology — Process assessment — Requirements for process measurement frameworks*, 3.7]

Note 1 to entry: The construct is a consequence of its measures and each measure is a determinant of the construct.

3.1646

forms design sheet

1. layout chart, intended as an aid for the placing of rules and other pre-printed matter in the designing of forms, containing margin indicators and a network of lines indicating the locations of printed rules [ISO 3535:1977 *Forms design sheet and layout chart*, 4.3]

3.1647

forward pass

1. a critical path method technique for calculating the early start and early finish dates by working forward through the schedule model from the project start date or a given point in time [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. schedule network analysis, backward pass

3.1648

forward recovery

1. reconstruction of a file to a given state by updating an earlier version, using data recorded in a chronological record of changes made to the file **2.** type of recovery in which a system, program, database, or other system resource is restored to a new, not previously occupied state in which it can perform required functions

3.1649

FOSS

1. free and open source software

3.1650

four-address instruction

1. computer instruction that contains four address fields
cf. one-address instruction, two-address instruction, three-address instruction, zero-address instruction

EXAMPLE: an instruction to add the contents of locations A, B, and C, and place the result in location D

3.1651

four-plus-one address instruction

1. computer instruction that contains five address fields, the fifth containing the address of the instruction to be executed next
cf. one-plus-one address instruction, two-plus-one address instruction, three-plus-one address instruction

EXAMPLE: an instruction to add the contents of locations A, B, and C, place the results in location D, then execute the instruction at location E

3.1652

fourth-generation language (4GL)

1. computer language designed to improve the productivity achieved by high-order (third-generation) languages and, often, to make computing power available to non-programmers
cf. machine language, assembly language, high order language, fifth-generation language

Note 1 to entry: Features typically include an integrated database management system, query language, report generator, screen definition facility, graphics generator, decision support function, financial modeling, spreadsheet capability, and statistical analysis functions.

3.1653

FP-EPA

1. fixed price with economic price adjustment [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1654

FPA

1. function point analysis [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 4; ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1655

FPA table

1. an entity type that has a secondary function in the application (e.g., code tables, reference tables, entity types with constants, text, or decodings) and whose data can be maintained by the application to be counted or by a different application [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1656

FPA tables EIF

- 1.** the external interface file that is counted for the set of all FPA tables identified in an application that are only used by the application to be counted, but that are maintained by a different application [*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*]

3.1657

FPA tables ILF

- 1.** the internal logical file that is counted for the set of all identifiable and maintainable FPA tables in an application [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1658

FPGA

- 1.** field programmable gate array [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.1659

FPIF

- 1.** fixed price incentive fee [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1660

FQR

- 1.** formal qualification review

3.1661

FRACAS

- 1.** failure reporting and corrective action system [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.1662

frame

- 1.** mechanism for dividing a browser window into independent windows for displaying different content or different parts of the same content (document) [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.10*]

3.1663

framework

- 1.** reusable design (models and/or code) that can be refined (specialized) and extended to provide some portion of the overall functionality of many applications. [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.64*] **2.** partially completed software subsystem that can be extended by appropriately instantiating some specific plug-ins

3.1664

free float

- 1.** the amount of time that a schedule activity can be delayed without delaying the early start date of any successor or violating a schedule constraint. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. total float

3.1665

freedom from risk

- 1.** degree to which a product or system mitigates the potential risk to economic status, human life, health, or the environment. [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.1.4*]

cf. risk

3.1666

front matter

1. material that comes at the front of a printed book or manual, such as the title page and table of contents

3.1667

frozen branch

1. branch where no development takes place, either in preparation for a release or because active development has ceased on it

3.1668

FRP

1. full-rate production. [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.1669

FRR

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

3.1670

FS

1. finish-to-start. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1671

FSM

1. functional size measurement. [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 4; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.11*]

3.1672

FSM method

1. a specific implementation of FSM defined by a set of rules. [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.4*]

3.1673

FSMM

1. functional size measurement method. [*ISO/IEC 14143-6:2012 Information technology — Software measurement — Functional size measurement — Part 6: Guide for use of ISO/IEC 14143 series and related International Standards, 2*]

3.1674

FTP

1. File Transfer Protocol. [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.1675

FTR

1. file type referenced. [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 4; ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1676

full duplex

1. able to communicate data in both directions simultaneously

cf. half duplex

3.1677

function

1. defined objective or characteristic action of a system or component**2.** software module that performs a specific action, is invoked by the appearance of its name in an expression, receives input values, and returns a single value

3. part of an application that provides facilities for users to carry out their tasks [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.21] 4. elementary unit of requirements and specifications defined and used for measurement purposes [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*] 5. single-valued mapping [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.65] 6. transformation of inputs to outputs, by means of some mechanisms, and subject to certain controls, that is identified by a function name and modeled by a box [IEEE 1320.1-1998 (R2004) *IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.53]

3.1678

function name

1. active verb or verb phrase that describes what is to be accomplished by a function [IEEE 1320.1-1998 (R2004) *IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.54]

Note 1 to entry: A box takes as its box name the function name of the function represented by the box.

3.1679

function point (FP)

1. a unit which expresses the size of an application or of a project [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. unit of measure for functional size as defined within ISO/IEC 20926:2009 [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.35] 3. a measure of the delivered software functionality [IEEE 1045-1992, (R2002) *IEEE Standard for Software Productivity Metrics*, 3.2]

3.1680

function point analysis (FPA)

1. a method used to acquire a measurement of the amount of functionality an application provides a user [ISO/IEC 24570:2005 *Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*] 2. a form of functional size measurement (FSM) that measures the functional size of software development, enhancement and maintenance activities associated with business applications, from the customer's point of view [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10] 3. method for measuring functional size as defined in ISO/IEC 29026:2009 [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.36]

3.1681

function point count

1. activity of applying rules to measure the functional size of an application or project [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.37] 2. the absolute sum of the number of function points of all the functions to be added to, changed in, or deleted from the project or the application to be counted [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*]

Note 1 to entry: Three types of function point count are application, development project, and enhancement project.

3.1682

function point table

1. a table used to allocate function points to functions, depending on the function type and the complexity established for the function [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*]

3.1683

function type

1. the five types of components of which an application consists, seen from the perspective of FPA [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. type of base functional component identified in ISO/IEC 20926:2009 [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.38]

Note 1 to entry: The five function types identified in ISO/IEC 20926:2009 are External Input, External Output, External Inquiry, Internal Logical File and External Interface File.

3.1684

function-oriented design

1. partitioning of a design into subsystems and modules, with each one handling one or more functions
cf. object-oriented design, data-structure-oriented design

3.1685

functional analysis

1. systematic investigation of the functions of a real or planned system [*ISO/IEC 2382:2015, Information technology — Vocabulary*] 2. examination of a defined function to identify all the subfunctions necessary to accomplish that function, to identify functional relationships and interfaces (internal and external) and capture these in a functional architecture, to flow down upper-level performance requirements and to assign these requirements to lower-level subfunctions

3.1686

functional appropriateness

1. degree to which the functions facilitate the accomplishment of specified tasks and objectives [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.1.3*]

EXAMPLE: A user is only presented with the necessary steps to complete a task, excluding any unnecessary steps.

Note 1 to entry: Functional appropriateness corresponds to suitability for the task.

3.1687

functional architecture

1. arrangement of functions and their subfunctions and interfaces (internal and external) that defines the execution sequencing, conditions for control or data flow, and the performance requirements to satisfy the requirements baseline 2. hierarchical arrangement of functions, their internal and external (external to the aggregation itself) functional interfaces and external physical interfaces, their respective functional and performance requirements, and their design constraints

3.1688

functional baseline

1. description of the system's performance (functional, interoperability, and interface characteristics) and the verification required to demonstrate the achievement of those specified characteristics [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1*]
cf. allocated baseline, developmental configuration, product baseline

3.1689

functional cohesion

1. type of cohesion in which the tasks performed by a software module all contribute to the performance of a single function
cf. coincidental cohesion, communicational cohesion, logical cohesion, procedural cohesion, sequential cohesion, temporal cohesion

3.1690

functional completeness

1. degree to which the set of functions covers all the specified tasks and user objectives [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.1.1*]

3.1691

functional complexity

1. specific complexity rating assigned to a function using defined rules [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.32*]

3.1692

functional configuration audit (FCA)

1. audit conducted to verify that the development of a configuration item has been completed satisfactorily, that the item has achieved the performance and functional characteristics specified in the functional or allocated configuration identification, and that its operational and support documents are complete and satisfactory [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]
cf. configuration management, physical configuration audit

3.1693

functional configuration identification

1. in configuration management, the current approved technical documentation for a configuration item
cf. allocated configuration identification, product configuration identification functional baseline

Note 1 to entry: It prescribes all necessary functional characteristics, the tests required to demonstrate achievement of specified functional characteristics, the necessary interface characteristics with associated configuration items, the configuration item's key functional characteristics and its key lower-level configuration items, if any, and design constraints.

3.1694

functional correctness

1. degree to which a product or system provides the correct results with the needed degree of precision [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.1.2*]

3.1695

functional decomposition

1. type of modular decomposition in which a system is broken down into components that correspond to system functions and subfunctions
cf. hierarchical decomposition, stepwise refinement

3.1696

functional design

1. process of defining the working relationships among the components of a system 2. result of the process in (1)
3. specification of the functions of the components of a system and of the working relationships among them [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. architectural design

3.1697

functional domain

1. class of software based on the characteristics of functional user requirements which are pertinent to FSM [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.5*] 2. categorized functions that are generally used together [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.16*]

3.1698

functional domain categorization (FDC)

1. a process for identifying functional domains that conforms to the requirements of ISO/IEC TR 14143-5:2004, 5.2. [*ISO/IEC TR 14143-5:2004 Information technology — Software measurement — Functional size measurement — Part 5: Determination of functional domains for use with functional size measurement, 3.2*]

3.1699

functional language

1. programming language used to express programs as a sequence of functions and function calls

EXAMPLE: LISP

3.1700

functional manager

line manager

1. someone with management authority over an organizational unit within a functional organization. The manager of any group that actually makes a product or performs a service. Sometimes called a line manager [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1701

functional organization

1. a hierarchical organization where each employee has one clear superior, and staff are grouped by areas of specialization and managed by a person with expertise in that area [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1702

functional process

functional process type

transactional process

1. elementary component of a set of functional user requirements, comprising a unique, cohesive and independently executable set of data movements [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.10*] **2.** an elementary component of a set of Functional User Requirements, comprising a unique, cohesive and independently executable set of data or data movements (functional services) [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, A.11*]

Note 1 to entry: It is triggered by a data movement (an Entry) from a functional user that informs the piece of software that the functional user has identified a triggering event, and is complete when it has executed all that is required to be done in response to the triggering event.

3.1703

functional product

1. product capable of performing computations [*ISO/IEC TR 14759:1999 Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use, 3.2 b*)]

3.1704

functional requirement

1. statement that identifies what results a product or process shall produce **2.** requirement that specifies a function that a system or system component shall perform [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*]

cf. nonfunctional requirement

3.1705

functional service

1. base functional component (BFC) [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, 3.6*] **2.** service that must be implemented in the piece of software in order to fulfill functional user requirements [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, A.9*]

3.1706

functional size

FS

1. size of the software derived by quantifying the functional user requirements [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.6; ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.33*]

3.1707

functional size measurement (FSM)

1. the process of measuring functional size [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.7; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.11*]

3.1708

functional size measurement method (FSMM)

FSM method

1. specific implementation of FSM defined by a set of rules [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.12*]

3.1709

functional specification

- 1.** document that specifies the functions that a system or component must perform

Note 1 to entry: often part of a requirements specification

3.1710

functional suitability

- 1.** degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.1*]

Note 1 to entry: Functional suitability is only concerned with whether the functions meet stated and implied needs, not the functional specification.

3.1711

functional system design

- 1.** specification of the functions of the components of a software system and of the working relationships between them [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.19*]

3.1712

functional testing

black-box testing

- 1.** testing that ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions **2.** testing conducted to evaluate the compliance of a system or component with specified functional requirements

3.1713

functional unit

- 1** entity of hardware or software, or both, capable of accomplishing a specified purpose [*ISO/IEC 2382:2015, Information technology—Vocabulary*]

3.1714

functional user

- 1.** user that is a sender or an intended recipient of data in the Functional User Requirements of a piece of software [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.13*]

3.1715

functional user requirements (FUR)

- 1.** subset of the user requirements specifying what the software shall do in terms of tasks and services [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.34*] **2.** a subset of the user requirements describing what the software does in terms of tasks and services [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.8; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.14*]

Note 1 to entry: Functional User Requirements include but are not limited to: data transfer (for example Input customer data, Send control signal); data transformation (for example Calculate bank interest, Derive average temperature); data storage (for example Store customer order, Record ambient temperature over time); data retrieval (for example List current employees, Retrieve aircraft position). User Requirements that are not Functional User Requirements include but are not limited to: quality constraints (for example usability, reliability, efficiency and portability); organizational constraints (for example locations for operation, target hardware and compliance to standards); environmental constraints (for example interoperability, security, privacy and safety); implementation constraints (for example development language, delivery schedule).

3.1716

functionality

- 1.** capabilities of the various computational, user interface, input, output, data management, and other features provided by a product

Note 1 to entry: This characteristic is concerned with what the software does to fulfill needs. The software quality characteristic functionality can be used to specify or evaluate the suitability, accuracy, interoperability, security, and compliance of a function.

3.1717

funding limit reconciliation

1. the process of comparing the planned expenditure of project funds against any limits on the commitment of funds for the project to identify any variances between the funding limits and the planned expenditures [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1718

FUR

1. functional user requirement(s) [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 4; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.14*]

3.1719

fuse ROM

1. programmable ROM with written data based on a fuse connection state

EXAMPLE: If the normal state of the fuse means logic "1", the breaking state is "0."

3.1720

future worth

1. representation of a cash flow as a single instance at the end of the planning horizon
cf. annual equivalent, present worth

3.1721

Gantt chart

1. a bar chart of schedule information where activities are listed on the vertical axis, dates are shown on the horizontal axis, and activity durations are shown as horizontal bars placed according to start and finish dates [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. bar chart

3.1722

garbage collection

1. in computer resource management, a synonym for memory compaction (1)

3.1723

GATES

1. Stage-Gate methodology [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*]

3.1724

general purpose input/output port (GPIO)

1. generic pin (port) on a microcomputer whose function (whether it is an input or output pin) is not predefined and is user-controlled

3.1725

general register

1. register that stores both addresses and data

3.1726

general system characteristics (GSCs)

1. terminology for technical complexity adjustment factors [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

3.1727

generality

1. degree to which a system or component performs a broad range of functions
cf. reusability

3.1728

generalization

- 1.** taxonomy in which instances of both entities represent the same real or abstract thing [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.66] cf. categorization

Note 1 to entry: One entity (the generic entity) represents the complete set of things and the other (category entity) represents a subtype or sub-classification of those things. The category entity can have one or more attributes, or relationships with instances of another entity, not shared by all generic entity instances. Each instance of the category entity is simultaneously an instance of the generic entity. [key style]

3.1729

generalization structure

- 1.** connection between a superclass and one of its more specific, immediate subclasses [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.69]

3.1730

generalization taxonomy

generalization hierarchy

generalization network

- 1.** set of generalization structures with a common generic ancestor [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.70]

Note 1 to entry: In a generalization taxonomy every instance is fully described by one or more of the classes in the taxonomy. The structuring of classes as a generalization taxonomy determines the inheritance of responsibilities among classes.

3.1731

generalize

- 1.** saying that a subclass s generalizes to a superclass C means that every instance of class s is also an instance of class C [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.66]

Note 1 to entry: Generalization is fundamentally different from a relationship, which can associate distinct instances.

3.1732

generally accepted

- 1.** knowledge to be included in the study material of a software engineering licensing exam that a graduate would pass after completing four years of work experience [*ISO/IEC TR 19759:2016, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOk)*]

3.1733

generated address

synthetic address

- 1.** address that has been calculated during the execution of a computer program
cf. absolute address, effective address, relative address, indirect address

3.1734

generation

- 1.** act of defining and describing a methodology from a particular metamodel [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 3.8]

Note 1 to entry: Generating a methodology includes explaining the structural position and semantics of each methodology element using the selected metamodel. Thus, what methodology elements are possible, and how they relate to each other, are constrained by such a metamodel. Usually, method engineers perform generation, yielding a complete and usable methodology.

3.1735

generic ancestor (of a class)

- 1.** superclass that is either an immediate superclass of the class or a generic ancestor of one of the superclasses of the class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.71]

cf. reflexive ancestor

3.1736

generic entity

1. entity whose instances are classified into one or more subtypes or subclassifications (category entities) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.72]

cf. superclass, supertype

Note 1 to entry: [key style]

3.1737

generic practice

1. activity that, when consistently performed, contributes to the achievement of a specific process attribute [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.3.6]

3.1738

generic profile group

1. profile group applicable to very small entities (VSEs) that do not develop critical systems or software products and have typical situational factors [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.28; *ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.9]

3.1739

generic program unit

1. software module that is defined in a general manner and that requires substitution of specific data, instructions, or both, in order to be used in a computer program

cf. instantiation

3.1740

GFE

1. government-furnished equipment [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.1741

GFI

1. government-furnished information [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes*, 3.2]

3.1742

GIF

1. General Interworking Framework [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions*, 4] **2.** Graphics Interchange Format [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.1743

GIOP

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

3.1744

glass box

white box

1. system or component whose internal contents or implementation are known **2.** pertaining to an approach that treats a system or component as in (1)

cf. black box

3.1745

global attribute

1. condition when the attributes that describe the foreign keys are the same attributes (and attribute values) as those describing the associated candidate key [*ISO/IEC 15476-4:2005 Information technology — CDIF semantic metamodel — Part 4: Data models*, 6.10]

3.1746

global compaction

1. in microprogramming, compaction in which microoperations can be moved beyond the boundaries of the single-entry, single-exit sequential blocks in which they occur
cf. local compaction

3.1747

global data

common data

1. data that can be accessed by two or more non-nested modules of computer program without being explicitly passed as parameters between the modules

cf. local data

3.1748

global label

1. label associated with the net graph itself, rather than with an object of a net graph [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.4]

3.1749

global navigation

1. set of navigation links available on all pages of a website [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.11]

3.1750

global variable

1. variable that can be accessed by two or more non-nested modules of a computer program without being explicitly passed as a parameter between the modules

cf. local variable

3.1751

globally unique identifier (GUID)

1. 16-byte string of characters that is generated in a manner that gives a high probability that the string is unique in any context [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.16]

Note 1 to entry: GUID as an all capitalized term refers specifically to the 16-byte version. If the term is in lowercase (guid), it refers to a general algorithm that can use either a URI or a 16-byte-based identifier.

3.1752

glossary

1. collection of the names and narrative descriptions of all terms that can be used for defined concepts within an environment [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.73] 2. set of definitions that includes arrow labels and box names used in an IDEF0 model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.55]

3.1753

glossary page

1. model page that contains definitions for the arrow labels and box names in a specific diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.56]

3.1754

go to

1. computer program statement that causes a jump
cf. call, case, if-then-else branch

3.1755

goal

- 1.** intended outcome [*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, 2.7*] **2.** intended outcome of user interaction with a product [*ISO/IEC 25062:2006 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports, 4.8*]

[SOURCE: ISO 9241-11]

3.1756

GOTS

- 1.** government-off-the-shelf.

3.1757

governance

- 1.** the process of establishing and enforcing strategic goals and objectives, organizational policies, and performance parameters [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.1758

Government-off-the-Shelf

GOTS

- 1.** software supplied by the government for reuse in another project
cf. COTS

3.1759

GPIO

- 1.** general-purpose input/output port

3.1760

grade

- 1.** a category or rank used to distinguish items that have the same functional use (e.g., "hammer"), but do not share the same requirements for quality (e.g., different hammers may need to withstand different amounts of force). [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.1761

granularity

- 1.** depth or level of detail at which data is collected

3.1762

graph

- 1.** diagram that represents the variation of a variable in comparison with that of one or more other variables **2.** diagram or other representation consisting of a finite set of nodes and internode connections called edges or arcs

EXAMPLE: graph showing a bathtub curve

3.1763

graphical information

- 1.** information defining the graphical appearance of objects and labels of a net graph, which can be the position, size, line color, fill color, font, or line width [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.3*]

3.1764

GraphicSymbol

- 1.** reference to a particular image or drawing for representing a particular software behavior concept instance [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.1765

Grosch's law

1. guideline formulated by H. R. J. Grosch, stating that the computing power of a computer increases proportionally to the square of the cost of the computer
cf. computer performance evaluation

3.1766

ground rules

1. list of acceptable and unacceptable behaviors adopted by a project team to improve working relationships, effectiveness, and communication **2.** expectations regarding acceptable behavior by project team members [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1767

group

1. number of model elements regarded as a unit formed by traceability relationships to a single distinct element [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.1768

group creativity techniques

1. techniques that are used to generate ideas within a group of stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1769

guarantee condition

1. statement of the constraints that will be satisfied by output interaction occurrences and the next property state as a result of the occurrence of a particular behavior pattern [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.13]

3.1770

guard condition

1. statement of the circumstances (input interaction occurrences and property state) that allow a stimulus to cause the occurrence of a particular behavior pattern [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.14]

3.1771

GUI

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

3.1772

GUID

1. globally unique identifier [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.16]

3.1773

guide

1. document published by ISO or IEC giving rules, orientation, advice or recommendations relating to international standardization [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.10]

cf. standard, guideline

Note 1 to entry: [ISO/IEC Directives, Part 2]

3.1774

guideline

1. an official recommendation or advice that indicates policies, standards, or procedures for how something should be accomplished [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. guide, standard

3.1775

gull wing lead

1. connector from the thin side of an integrated circuit package which extends out, down, and then out horizontally to allow it to be connected within a device

Note 1 to entry: The bent shape is thought to resemble a bird wing.

3.1776

hacker

1. technically sophisticated computer enthusiast [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** technically sophisticated computer enthusiast who uses his or her knowledge and means to gain unauthorized access to protected resources [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1777

half duplex

1. able to communicate data in both directions, but in only one direction at a time [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] *cf.* full duplex

3.1778

halt

1. most commonly, a synonym for stop **2.** less commonly, a synonym for pause **3.** [HALT] highly accelerated life testing [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.1779

hard copy

1. permanent copy of a display image generated on an output unit such as a printer or a plotter, and which can be carried away [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1780

hard failure

1. failure that results in complete shutdown of a system

cf. soft failure

3.1781

hardware

1. physical equipment used to process, store, or transmit computer programs or data **2.** all or part of the physical components of an information system [*ISO/IEC 2382:2015, Information technology — Vocabulary*] *cf.* software

3.1782

hardware configuration item (HCI)

1. aggregation of hardware that is designated for configuration management and treated as a single entity in the configuration management process

cf. software configuration item

Note 1 to entry: An HCI exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the hardware has been established as a configuration item.

3.1783

hardware description language (HDL)

hardware design language

1. software programming language used to design and model hardware, especially digital logic circuits

EXAMPLE: VHDL (IEEE 1076), Verilog (IEEE 1364)

3.1784

hardware engineering

1. application of a systematic, disciplined, and quantifiable approach to design, implement, and maintain a tangible product by transforming a set of requirements that represent the collection of stakeholder needs, expectations, and constraints; using documented techniques and technology

cf. software engineering, systems engineering

3.1785

hardware monitor

1. device that measures or records specified events or characteristics of a computer system **2.** a software tool that records or analyzes hardware events during the execution of a computer program

EXAMPLE: a device that counts the occurrences of various electrical events or measures the time between such events

3.1786

Harvard architecture

1. computer architecture with physically separate communication paths for instructions and data

3.1787

hazard

1. intrinsic property or condition that has the potential to cause harm or damage [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.11*] **2.** source of potential harm or a situation with a potential for harm in terms of human injury; damage to health, property, or the environment; or some combination of these [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.11*]

3.1788

hazard identification

1. process of recognizing that a hazard exists and defining its characteristics [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.12*]

3.1789

HCI

HWCI

1. human computer interface [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*] **2.** hardware configuration item

3.1790

HDD

1. hardware design description [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.2*]

3.1791

HDL

1. hardware description language **2.** hardware design language

cf. design language

3.1792

HDTV

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

3.1793

head

1. forefront of a branch, which contains the evolving versions of the source tree

Note 1 to entry: A release coming out of head will have the newest features but will also likely be unstable.

3.1794

head action

1. in a given activity, an action that has no predecessor [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.1.7*]

3.1795

header

1. block of comments placed at the beginning of a computer program or routine **2.** identification or control information placed at the beginning of a file or message **3.** material repeated at the top of each page

3.1796

heading

1. text that identifies the topic that will be covered in the following text

3.1797

health and safety risk mitigation

1. degree to which a product or system mitigates the potential risk to people in the intended contexts of use [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.4.2]

3.1798

heavyweight process

1. process with its own memory and multiple threads of control

3.1799

help system

1. ancillary part of a program, or sometimes a separate program, that allows the user to view parts of the online documentation or help text on request

cf. online documentation system

3.1800

hidden

1. both private and protected [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.74]

cf. public, private, protected

3.1801

hierarchical decomposition

1. type of modular decomposition in which a system is broken down into a hierarchy of components through a series of top-down refinements

cf. functional decomposition, stepwise refinement

3.1802

hierarchical modeling

1. technique used in computer performance evaluation, in which a computer system is represented as a hierarchy of subsystems, the subsystems are analyzed to determine their performance characteristics, and the results are used to evaluate the performance of the overall system

3.1803

hierarchically consecutive

1. unbroken unidirectional traversal of all nodes between two specified nodes in a tree [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.57]

3.1804

hierarchy

1. structure in which components are ranked into levels of subordination; each component has zero, one, or more subordinates; and no component has more than one superordinate component **2.** arrangement of model elements according to traceability relationships, where an element that owns or groups other elements is considered at a higher level than the owned (grouped) elements [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

cf. hierarchical decomposition, hierarchical modeling

3.1805

high level

1. general; abstract

3.1806

high-level design

1. process of defining the high-level concepts that guide low-level design and implementation

cf. architecture

Note 1 to entry: High-level design typically involves organizing a system into subprograms and specifying the interfaces between them.

3.1807

high-level net

1. algebraic structure comprising a set of places; a set of transitions; a set of types; a function associating a type to each place, and a set of modes (a type) to each transition; pre-function imposing token demands (multisets of tokens) on places for each transition mode; post function determining output tokens (multisets of tokens) for places for each transition mode; and an initial marking [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.11]

3.1808

high-level Petri Net graph

1. net graph and its associated annotations comprising place types, arc annotations and transition conditions, and their corresponding definitions in a set of declarations, and an initial marking of the net [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.12]

3.1809

high-order language (HOL)

high-level language

higher order language

third-generation language

1. programming language that requires little knowledge of the computer on which a program will run, can be translated into several different machine languages, allows symbolic naming of operations and addresses, provides features designed to facilitate expression of data structures and program logic, and usually results in several machine instructions for each program statement

cf. assembly language, fifth-generation language, fourth-generation language, machine language

EXAMPLE: Ada, COBOL, FORTRAN, ALGOL, PASCAL

3.1810

higher-level management

1. person or persons who provide the policy and overall guidance for the process, but do not provide the direct day-to-day monitoring and controlling of the process

Note 1 to entry: Such persons belong to a level of management in the organization above the immediate level responsible for the process.

3.1811

histogram

1. a special form of bar chart used to describe the central tendency, dispersion, and shape of a statistical distribution [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1812

historical information

1. documents and data on prior projects including project files, records, correspondence, closed contracts, and closed projects [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1813

HLL

1. high-level language

cf. high-order language

3.1814

HLPN

1. High-level Petri Net [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.2.1*]

3.1815

HLPNG

1. High-level Petri Net Graph [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.2.2*]

3.1816

HLPNS

1. High-level Petri Net Schema [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.2.3*]

3.1817

HMI

1. human-machine interface

cf. user interface

3.1818

HOL

1. high-order language

3.1819

home page

center page

front page

index page

main page

start page

top page

1. page of a website through which users typically enter the website, and whose URL is typically published or linked as the main web address of the site or organization [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.12*]

3.1820

homogeneous redundancy

1. in fault tolerance, realization of the same function with identical means

cf. diversity

EXAMPLE: use of two identical processors

3.1821

horizontal microinstruction

1. microinstruction that specifies a set of simultaneous operations needed to carry out a given machine language instruction

cf. diagonal microinstruction, vertical microinstruction

Note 1 to entry: Horizontal microinstructions are relatively long, often 64 bits or more, and are called 'horizontal' because the set of simultaneous operations that they specify are written on a single line, rather than being listed sequentially down the page.

3.1822

host machine

1. computer on which a program or file is installed **2.** in a computer network, a computer that provides processing capabilities to users of the network **3.** computer used to develop software intended for another computer **4.** computer used to emulate another computer

3.1823

hostile bailout

- 1.** bailout done without prior arrangement by a committer other than the one who introduced the original change

Note 1 to entry: This is usually the opening shot in a commit war.

3.1824

housekeeping operation

overhead operation

- 1.** computer operation that establishes or reestablishes a set of initial conditions to facilitate the execution of a computer program

EXAMPLE: initializing storage areas, clearing flags, rewinding tapes, opening and closing files

3.1825

HREF

- 1.** HTML reference designator [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.1826

HRS

- 1.** hardware requirements specification [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*]

3.1827

HSI

- 1.** human systems integration [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.2*]

3.1828

HTML

- 1.** HyperText Markup Language [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.1829

HTTP

- 1.** HyperText Transfer Protocol [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.1830

human resource management plan

human resource plan

- 1.** a component of the project or program management plan that describes how the roles and responsibilities, reporting relationships and staff management will be addressed and structured [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1831

human resource planning

- 1.** identification and documentation of project roles, responsibilities and reporting relationships, as well as estimation of required staff by time period and creation of a staffing management plan

3.1832

human systems engineering

- 1.** activities involved throughout the system life cycle that address the human element of system design (including usability, measures of effectiveness, measures of performance, and total ownership cost)

Note 1 to entry: These activities include the definition and synthesis of manpower, personnel, training, human engineering, health hazards, and safety issues.

3.1833

human systems integration (HSI)

1. interdisciplinary technical and management process for integrating human considerations with and across all system elements, an essential enabler to systems engineering practice [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.11]

3.1834

human-centered design

user-centered design

1. approach to system design and development that aims to make interactive systems more usable by focusing on the use of the system; applying human factors, ergonomics and usability knowledge and techniques [*ISO/IEC 25063:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description*, 3.6]

Note 1 to entry: The term "human-centered design" is used rather than "user-centered design" to emphasize that design impacts a number of stakeholders, not just those typically considered as users. However, in practice, these terms are often used synonymously. Usable systems can provide a number of benefits including improved productivity, enhanced user well-being, avoidance of stress, increased accessibility, and reduced risk of harm.

3.1835

Hurvitz criterion

1. in decision making under uncertainty, a method which gives each decision a value which is a weighted sum of its worst and best possible outcomes

cf. maximax rule, maximin rule, minimax regret rule

Note 1 to entry: allows the decision maker to account for optimistic and pessimistic views

3.1836

HW

1. hardware [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]

3.1837

HWCI

1. hardware configuration item [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.1838

hybrid computer

1. computer that integrates analog computer components and digital computer components by interconnection of digital-to-analog converters and analog-to-digital converters [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: A hybrid computer can use or produce analog data and discrete data.

3.1839

hybrid coupling

1. type of coupling in which different subsets of the range of values that a data item can assume are used for different and unrelated purposes in different software modules

cf. common-environment coupling, content coupling, control coupling, data coupling, pathological coupling

3.1840

I/O

1. input/output

3.1841

I/O task-structuring criteria

1. category of the task-structuring criteria addressing how device interface objects are mapped to I/O tasks and when an I/O task is activated

3.1842

I2C

IIC

1. inter-integrated circuit bus

3.1843

IAP

1. in-application programming

3.1844

IBa

1. issue benchmarks activity [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*, 4]

3.1845

IBD

1. information-based domain [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 4.2]

3.1846

ICD

1. initial capabilities document [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.1847

ICE

1. in-circuit emulator

3.1848

ICOM

1. input, control, output, and mechanism [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.2]

3.1849

ICOM code

arrow reference

1. expression in one diagram that unambiguously identifies an arrow segment in another diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.58]

Note 1 to entry: An ICOM code is used to associate a boundary arrow of a child diagram with an arrow attached to an ancestral box.

3.1850

ICOM label

1. arrow label attached without a squiggle directly to the arrowhead of an output boundary arrow or to the arrowtail of an input, control, or mechanism boundary arrow [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.59]

Note 1 to entry: An ICOM label associates a boundary arrow of a child diagram with an arrow label of an arrow attached to an ancestral box.

3.1851

icon

1. graphic displayed on the screen that represents a function of the computer system [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.22]

3.1852

ICS

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

3.1853

ICT

1. information and communication technology [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.1854

ICWG

1. Interface Control Working Group [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 5]

Note 1 to entry: Depending on the size and complexity of a project, can be a group of people, a single person or a function

3.1855

IDD

1. interface design document [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]

3.1856

IDE

1. integrated development environment

3.1857

idea/mind mapping

idea mapping

mind mapping

1. technique used to consolidate ideas created through individual brainstorming sessions into a single map to reflect commonality and differences in understanding, and to generate new ideas [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1858

ideal time

1. a best-case estimate of the time needed for a developer or team to complete a task or deliver a feature [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.1859

IDEF0 model

1. abstractly, a hierarchical set of IDEF0 diagrams that depict, for a specific purpose and from a specific viewpoint, the functions of a system or subject area, along with supporting glossary, text, and For Exposition Only (FEO) information [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.60]

Note 1 to entry: Concretely, a set of model pages that include at least an A-0 context diagram and an A0 decomposition diagram, a glossary or specific glossary pages, one or more text pages to accompany each diagram, and FEO pages and model pages of other types as needed.

3.1860

IDEF1X model

1. et of one or more IDEF1X views, often represented as view diagrams that depict the underlying semantics of the views, along with definitions of the concepts used in the views [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.75]

3.1861

identifier

1. unambiguous name, in a given naming context [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 12.2] **2.** name, address, label, or distinguishing index of an object in a computer program **3.** within an IDEF0 model, a model name, a box name, or an arrow label [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.61]

3.1862

identify risks

- 1.** the process of determining which risks may affect the project and documenting their characteristics [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1863

identify stakeholders

- 1.** the process of identifying the people, groups or organizations that could impact or be impacted by a decision, activity or outcome of the project, analyzing and documenting relevant information regarding their interests, involvement, interdependencies, influence, and potential impact on project success [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1864

identifying relationship

- 1.** specific (not many-to-many) relationship in which every attribute in the primary key of the parent entity is contained in the primary key of the child entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.79*]

cf. nonidentifying relationship [key style]

3.1865

identity

- 1.** inherent property of an instance that distinguishes it from all other instances [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.80*]

Note 1 to entry: Identity is intrinsic to the instance and independent of the instance's property values or the classes to which the instance belongs.

3.1866

identity-style view

- 1.** view produced using the identity-style modeling constructs [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.81*]

3.1867

IDL

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

3.1868

idle

- 1.** pertaining to a system or component that is operational and in service, but not in use
cf. busy, down, up

3.1869

idle time

standby time

- 1.** period of time during which a system or component is operational and in service, but not in use
cf. busy time, down time, set-up time, up time

3.1870

IEC

- 1.** International Electrotechnical Commission [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 5*]

3.1871

IEEE

- 1.** Institute of Electrical and Electronics Engineers [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*]

3.1872

IETF

- 1.** Internet Engineering Task Force

3.1873

if-then-else

1. single-entry, single-exit two-way branch that defines a condition, specifies the processing to be performed if the condition is met and, optionally, if it is not, and returns control in both instances to the statement immediately following the overall construct

cf. case, jump, go to, dyadic selective construct, monadic selective construct

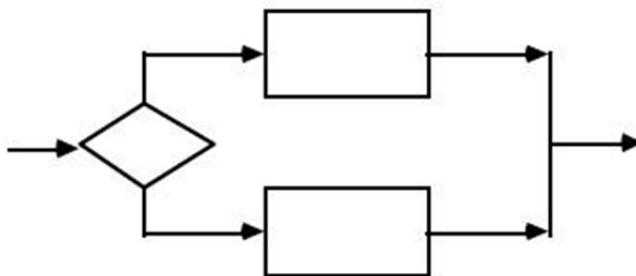


Figure 14 — If-then-else construct

3.1874

IFB

1. invitation for bid [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1875

iff

1. if and only if [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.2.4]

3.1876

IFPUG

1. International Function Point Users Group [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*]

Note 1 to entry: membership governed, non-profit organization committed to promoting and supporting function point analysis and other software measurement techniques. The IFPUG maintains the definition of the direct descendent of the Albrecht 1984 FPA method

3.1877

IOP

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

3.1878

IOP-IOR

1. Internet Inter-ORB Protocol — Interoperable Object Reference [*ISO/IEC 14753:1999 Information technology — Open Distributed Processing — Interface references and binding*, 4]

3.1879

ILF

1. internal logical file [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 4; *ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1880

illustration

1. graphic element set apart from the main body of text and normally cited within the main text. [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.23]

Note 1 to entry: used as the generic term for tables, figures, exhibits, screen captures, flow charts, diagrams, drawings, icons, and other graphic elements

3.1881

illustrative product

- 1.** non-functional product [*ISO/IEC TR 14759:1999 Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use, 3.2 c]*]

3.1882

image processing

- 1.** use of a data processing system to create, scan, analyze, enhance, interpret, or display images [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1883

immediate data

- 1.** data contained in the address field of a computer instruction
cf. direct address, indirect address, n-level address, immediate instruction

3.1884

immediate instruction

- 1.** computer instruction whose address fields contain the values of the operands rather than the operands' addresses

cf. direct instruction, indirect instruction, absolute instruction, effective instruction, immediate data

3.1885

immunity

- 1.** degree to which a product or system is resistant to attack [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.6*]

cf. integrity

3.1886

immutable class

- 1.** class for which the set of instances is fixed; its instances do not come and go over time [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.82*]
cf. mutable class, value class

3.1887

IMP

- 1.** integrated master plan [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.1888

impact analysis

- 1.** identification of all system and software products that a change request affects and development of an estimate of the resources needed to accomplish the change

Note 1 to entry: This includes determining the scope of the changes to plan and implement work, accurately estimating the resources needed to perform the work, and analyzing the requested changes' cost and benefits.

3.1889

imperative construct

- 1.** sequence of one or more steps not involving branching or iteration

3.1890

implementable standard

- 1.** template for a technology object [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 9.1.1*]

3.1891

implementation

- 1.** process of translating a design into hardware components, software components, or both **2.** result of the process in (1)**3.** definition that provides the information needed to create an object and allow the object to participate in providing an appropriate set of services. [ISO/IEC 19500-2:2012 *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.8] **4.** installation and customization of packaged software **5.** construction **6.** system development phase at the end of which the hardware, software and procedures of the system become operational [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **7.** process of instantiation whose validity can be subject to test [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 9.1.2] **8.** (documentation) activity during which user documentation is created according to the design, tested, and revised [ISO/IEC/IEEE 26512:2011 *Systems and software engineering — Requirements for acquirers and suppliers of user documentation*, 4.16] **9.** software life cycle process that contains activities of requirements analysis, design, coding, integration, testing, installation, and support for acceptance of software products [ISO/IEC 90003:2014 *Software engineering — Guidelines for the application of ISO 9001:2008 to computer software*, 3.5]
cf. coding

3.1892

implementation phase

- 1.** period of time in the software life cycle during which a software product is created from design documentation and debugged

3.1893

implementation requirement

- 1.** requirement that specifies or constrains the coding or construction of a system or system component
cf. design requirement, functional requirement, interface requirement, performance requirement, physical requirement

3.1894

implementer

- 1.** organization that performs implementation tasks [ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes*, 4.15]
cf. developer

3.1895

implied addressing

- 1.** method of addressing in which the operation field of a computer instruction implies the address of the operands
cf. direct address, indirect address, one-ahead addressing, relative address, repetitive addressing

EXAMPLE: If a computer has only one accumulator, an instruction that refers to the accumulator needs no address information describing it.

3.1896

implied needs

- 1.** needs that have not been stated but are actual needs [ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.12]

EXAMPLE: needs not stated but implied by other stated needs and needs not stated because they are considered to be evident or obvious.

Note 1 to entry: Some implied needs only become evident when the system or software product is used in particular conditions.

3.1897

import process

- 1.** process of incorporating the content of a transfer file into a target environment [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.1]

3.1898

importer

- 1.** agent of the import process [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.1]

3.1899

imposed date

- 1.** a fixed date imposed on a schedule activity or schedule milestone, usually in the form of a "start no earlier than" and "finish no later than" date [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1900

impossible zone

- 1.** in a range of estimates, the region that is impossible under any circumstances to achieve

Note 1 to entry: For example, it is impossible to drive a car 500 miles in less than one hour, so the one-hour outcome for a 500-mile car trip is in the impossible zone for the estimate of how long it will take to drive 500 miles.

3.1901

improvability

- 1.** inherent ability of an organization to support continual process improvement [*ISO/IEC TR 33014:2013 Information technology — Process assessment — Guide for process improvement*, 3.3]

3.1902

IMS

- 1.** integrated master schedule [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.1903

in-application programming (IAP)

- 1.** capability of a microcontroller unit to fetch new program code and reprogram itself while the system is operating
cf. in-system programming

3.1904

in-circuit emulator (ICE)

- 1.** hardware device used to debug the software of an embedded system

3.1905

in-system programming (ISP)

- 1.** capability of a microcontroller unit to allow the user to download new code (reprogram the unit), activated by restarting the unit
cf. in-application programming

incentive fee

- 1.** a set of financial incentives related to cost, schedule, or technical performance of the seller [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1906

incident

- 1.** anomalous or unexpected event, set of events, condition, or situation at any time during the life cycle of a project, product, service, or system [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.22; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.21] **2.** unplanned interruption to a service or a reduction in the quality of a service at a specific time
cf. software test incident

3.1907

incident report

1. documentation of the occurrence, nature, and status of an incident [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.18]
cf. test incident report

3.1908

incipient failure

1. failure that is about to occur

3.1909

include

1. in UML, a relationship from a base use case to an included use case specifying how the behavior defined for the included use case can be inserted into the behavior defined for the base use case **2.** [information] having either the information or a reference to the information present in the document [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.12]

3.1910

income function

1. objective function that characterizes the income generated by different values of the decision variable
cf. cost function

3.1911

incomplete process

1. process that is not performed or is performed only partially

Note 1 to entry: One or more of the specific goals of the process are not satisfied.

3.1912

inconsistency ratio

1. in analytic hierarchy process (AHP), a function that measures how consistently the decision analyst assigned the values to the pair-wise comparisons

3.1913

increment

1. a tested, deliverable version of a software product that provides new or modified capabilities [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.1914

incremental analysis

1. consideration of the relative differences between alternatives

Note 1 to entry: If the incremental benefit of a second alternative over the first is more than the incremental investment between them, the second alternative is a better investment than the first.

3.1915

incremental benefit

1. additional income from one alternative compared to another

Note 1 to entry: If Alternative A generates \$10,000 and Alternative B generates \$12,000, the incremental benefit between A and B is \$2 000.

3.1916

incremental compiler

conversational compiler

interactive compiler

online compiler

1. compiler that completes as much of the translation of each source statement as possible during the input or scanning of the source statement

Note 1 to entry: Typically used for online computer program development and checkout.

3.1917

incremental development

1. software development technique in which requirements definition, design, implementation, and testing occur in an overlapping, iterative (rather than sequential) manner, resulting in incremental completion of the overall software product

cf. waterfall model, data structure-centered design, input-process-output, modular decomposition, object-oriented design

3.1918

incremental investment

1. avoidable additional investment between one alternative and another

Note 1 to entry: If Alternative A costs \$10 000 and Alternative B costs \$12 000, the incremental investment between A and B is \$2000.

3.1919

incremental life cycle

1. a project life cycle where the project scope is generally determined early in the project lifecycle, but time and cost estimates are routinely modified as the project team understanding of the product increases. Iterations develop the product through a series of repeated cycles, while increments successively add to the functionality of the product [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. iterative life cycle

3.1920

incremental productivity

1. productivity computed periodically during development

3.1921

independence

1. of software quality assurance (SQA), situation in which SQA is free from technical, managerial, and financial influences, intentional or unintentional [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*]

3.1922

independent

1. performed by an organization free from control by the supplier, developer, operator, or maintainer

3.1923

independent entity

identifier-independent entity

1. entity for which each instance can be uniquely identified without determining its relationship to another entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.84*]

cf. dependent entity [key style]

3.1924

independent estimates

1. a process of using a third party to obtain and analyze information to support prediction of cost, schedule or other items [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1925

independent evaluator

1. individual or organization that performs an evaluation independently from developers and acquirers [*ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.30*]

Note 1 to entry: The individual or organization acting as developer or acquirer for the target system to be evaluated cannot become the independent evaluator for the system. The independent evaluator can be an organization.

3.1926

independent state class

- 1.** state class that is not a dependent state class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.85]
cf. dependent state class

3.1927

independent verification and validation (IV&V)

- 1.** verification and validation performed by an organization that is technically, managerially, and financially independent of the development organization [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1]

3.1928

indexed address

- 1.** address that must be added to the contents of an index register to obtain the address of the storage location to be accessed
cf. offset (2), relative address, self-relative address

3.1929

indicative function point count

- 1.** an indication denoting the estimated size of an application or project, based exclusively on a conceptual data model or a data model in the third normal-form [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.1930

indicator

- 1.** measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.10; *ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.13] **2.** device or variable that can be set to a prescribed state based on the results of a process or the occurrence of a specified condition

EXAMPLE: a flag or semaphore

3.1931

indicator value

- 1.** numerical or categorical result assigned to an indicator [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.11]

3.1932

indigenous error

- 1.** computer program error that has not been purposely inserted as part of an error-seeding process

3.1933

indirect address

multilevel address

- 1.** address that identifies the storage location of another address
cf. direct address, immediate data, indirect instruction, n-level address

Note 1 to entry: The designated storage location can contain the address of the desired operand or another indirect address; the chain of addresses eventually leads to the operand.

3.1934

indirect instruction

- 1.** computer instruction that contains indirect addresses for its operands
cf. direct instruction, immediate instruction, absolute instruction, effective instruction

3.1935

indirect labor

- 1.** human effort that is not directly associated with the units being produced

cf. direct labor

3.1936

indirect measure

- 1.** measure of an attribute that is derived from measures of one or more other attributes

Note 1 to entry: An external measure of an attribute of a computing system (such as the response time to user input) is an indirect measure of attributes of the software as the measure will be influenced by attributes of the computing environment as well as attributes of the software.

3.1937

indirect user

- 1.** person who receives output from a system, but does not interact with the system [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.6]

cf. direct user, secondary user

3.1938

inductive assertion method

- 1.** a proof of correctness technique in which assertions are written describing program inputs, outputs, and intermediate conditions, a set of theorems is developed relating satisfaction of the input assertions to satisfaction of the output assertions, and the theorems are proved or disproved using proof by induction [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual*]

3.1939

infant mortality

- 1.** set of failures that occur during the early-failure period of a system or component

3.1940

influence diagram

- 1.** a graphical representation of situations showing causal influences, time ordering of events, and other relationships among variables and outcomes [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1941

influencer

- 1.** persons or groups that are not directly related to the acquisition or use of the product, but, who can affect the course of the project, positively or negatively, due to their position in the customer organization

cf. stakeholder

3.1942

informal testing

- 1.** testing conducted in accordance with test plans and procedures that have not been reviewed and approved by a customer, user, or designated level of management

cf. formal testing

3.1943

information

- 1.** knowledge that is exchangeable amongst users, about things, facts, concepts, and so on, in a universe of discourse [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 3.2.6] **2.** In information processing, knowledge concerning objects, such as facts, events, things, processes, or ideas, including concepts, that within a certain context has a particular meaning [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: Although information will necessarily have a representation form to make it communicable, it is the interpretation of this representation (the meaning) that is relevant in the first place.

3.1944

information analysis

- 1.** systematic investigation of information and its flow in a real or planned system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1945

information architecture

- 1.** (human-centered) structure of an information space and the semantics for accessing required task objects, system objects and other information [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.8*]

Note 1 to entry: The appropriate combination of organization, labeling, navigation schemes and retrieval mechanisms within an information space will facilitate task completion and efficient access to content.

3.1946

information content

- 1.** set of metamodel and model instances found in a CDIF transfer [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

3.1947

information design

- 1.** process of developing content that meets the needs of the audience [*ISO/IEC/IEEE 26511:2011 Systems and software engineering — Requirements for managers of user documentation, 4.11*]

3.1948

information gathering techniques

- 1.** repeatable processes used to assemble and organize data across a spectrum of sources [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1949

information hiding

- 1.** software development technique in which each module's interfaces reveal as little as possible about the module's inner workings and other modules are prevented from using information about the module that is not in the module's interface specification**2.** containment of a design or implementation decision in a single module so that the decision is hidden from other modules
cf. encapsulation

3.1950

information item

- 1.** separately identifiable body of information that is produced, stored, and delivered for human use [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.3; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.23; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.22; ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.11*] **2.** separately identifiable body of information that is produced and stored for human use during a system or software life cycle [*ISO/IEC 25063:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description*]
cf. information product

Note 1 to entry: An information item can be produced in several versions during a system, software, or service life cycle.

3.1951

information item content

- 1.** information included in an information item, associated with a system, product or service, to satisfy a requirement or need [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.13*]

3.1952

information item type

generic document type

- 1.** group of information items consistent with a pre-arranged set of generic criteria [ISO/IEC/IEEE 15289:2015 *Systems and software engineering — Content of life-cycle information products (documentation), 5.14*]

EXAMPLE: A 'plan' is the information item type for all plans and 'report' is the information item type for all reports.

3.1953

information management

- 1.** in an information processing system, the functions of controlling the acquisition, analysis, retention, retrieval, and distribution of information [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.1954

information management system

- 1.** facilities, processes and procedures used to collect, store and distribute information between producers and consumers of information in physical or electronic format [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1955

information need

- 1.** insight necessary to manage objectives, goals, risks, and problems [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process, 3.12; ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.14*]

3.1956

information processing

- 1.** systematic performance of operations upon information, which includes data processing and can include operations such as data communication and office automation [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

Note 1 to entry: The term information processing is not a synonym for data processing

3.1957

information processing requirements

- 1.** the set of functions required by the commissioning user of the application software product (excluding any technical and quality requirements) [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 1.1*] cf. software

3.1958

information processing system

- 1.** one or more data processing systems and devices, such as office and communication equipment, that perform information processing [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.1959

information product

- 1.** one or more indicators and their associated interpretations that address an information need [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process, 3.13*] cf. information item

EXAMPLE: a comparison of a measured defect rate to planned defect rate along with an assessment of whether or not the difference indicates a problem

3.1960

information provisioning

- 1.** collection of all the infrastructure tools, software applications, non-automated elements, data sets, user documentation, and organizational structures which serve to supply information to the business [ISO/IEC 16350-2015 *Information technology — Systems and software engineering — Application management, 4.20*]

3.1961

information radiator

1. a large and frequently updated display of project information that is continually visible to the project team and other stakeholders [*Software Extension to the PMBOK® Guide Fifth Edition*]

EXAMPLE: burndown charts, cumulative flow diagrams, parking lot diagrams

3.1962

information retrieval (IR)

1. actions, methods, and procedures for obtaining information on a given subject from stored data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.1963

information security

1. preservation of confidentiality, integrity and accessibility of information

Note 1 to entry: In addition, other properties such as authenticity, accountability, non-repudiation and reliability can also be involved.

3.1964

information security incident

1. single or a series of unwanted or unexpected information security events that have a significant probability of compromising business operations and threatening information security

Note 1 to entry: [ISO/IEC 27000:2009]

3.1965

information system

1. information processing system, together with associated organizational resources such as human, technical, and financial resources, which provides and distributes information [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** all of the functions (input, output, transport, processing, and storage) of an application, databases, technical facilities, and manual procedures which support business processes [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.21] **3.** one or more computer systems and communication systems together with associated organizational resources such as human, technical, and financial resources that provide and distribute information [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.24] **c.** application

3.1966

information technology

IT

1. resources required to acquire, process, store and disseminate information

Note 1 to entry: includes Communication Technology (CT) and the composite term Information and Communication Technology (ICT)

3.1967

information technology project

IT project

1. temporary endeavor undertaken to create or change a unique information technology product, system, or service [*ISO/IEC 29155-1:2011 Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.7]

3.1968

information viewpoint

1. viewpoint on an ODP system and its environment that focuses on the semantics of information and information processing [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.1.1.2]

3.1969

Information-based domain (IBD)

- 1.** realm of activity for which information is the most valuable asset [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies, 3.1*]

Note 1 to entry: Information creation, manipulation, and dissemination are the most important activities within information-based domains. Typical information-based domains are software and systems engineering, business process reengineering, and knowledge management

3.1970

infrastructure

ecosystem

- 1.** hardware and software environment to support computer system and software design, development, and modification

3.1971

inheritance

- 1.** semantic notion by which the responsibilities (properties and constraints) of a subclass are considered to include the responsibilities of a superclass, in addition to its own, specifically declared responsibilities [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.86*]

3.1972

inherited attribute

- 1.** attribute that is a characteristic of a class by virtue of being an attribute of a generic ancestor [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.87*]
2. attribute that is a characteristic of a category entity by virtue of being an attribute in its generic entity or a generic ancestor entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.87*]

3.1973

inherited error

- 1.** error carried forward from a previous step in a sequential process

3.1974

initial Ent

initial entitlement schema

- 1.** Ent that is referenced by later Ents [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.14*]

Note 1 to entry: The initial Ent is typically a record of the first transaction between software licensor and end customer. An initial Ent is a type of primary Ent.

3.1975

initial function point count

- 1.** a function point count carried out at the beginning of a project [*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*]

3.1976

initial investment

first cost

- 1.** investment required just to start an activity

3.1977

initial marking (of the net)

- 1.** set of initial place markings given with the high-level net definition [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.14.1*]

3.1978

initial marking of a place

1. special marking of a place, defined with the high-level net [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.14.2]

3.1979

initial program loader

1. bootstrap loader used to load that part of an operating system needed to load the remainder of the operating system

3.1980

initial risk

1. estimated risk before applying risk reduction measures [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.6]

3.1981

initialization section

1. optional list of unconditional actions to be executed sequentially before the first condition is examined [*ISO 5806:1984 Information processing — Specification of single-hit decision tables*, 3.13]

Note 1 to entry: It can be written in the row which follows that of the table heading.

3.1982

initialize

1. to set a variable, register, or other storage location to a starting value
cf. clear, reset

3.1983

initiating object

1. object causing a communication [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.4.1]

3.1984

initiating process group

1. those processes performed to define a new project or a new phase of an existing project by obtaining authorization to start the project or phase [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.1985

initiator

1. person or organization that has both the ability and authority to start a project

3.1986

injection slot

1. point where the recoverability of the system under test (SUT) is tested by injecting a disturbance while a workload is being run [*ISO/IEC 25045:2010 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability*, 4.3]

3.1987

inline code

1. sequence of computer instructions that is physically contiguous with the instructions that logically precede and follow it

3.1988

inner cardinality

1. number of allowed instances of the relationship from the viewpoint of a single instance of the data object planning a role [*ISO/IEC 15476-4:2005 Information technology — CDIF semantic metamodel — Part 4: Data models*, 6.6.2]
cf. outer cardinality

3.1989

input

- 1.** data received from an external source **2.** pertaining to a device, process, or channel involved in receiving data from an external source **3.** to receive data from an external source **4.** to provide data from an external source **5.** loosely, input data **6.** in an IDEF0 model, that which is transformed by a function into output [IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.62] **7.** any item, whether internal or external to the project that is required by a process before that process proceeds. May be an output from a predecessor process [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition] **8.** data entered into an information processing system or any of its parts for storage or processing [ISO/IEC 2382:2015, Information technology — Vocabulary] **9.** process of entering data into an information-processing system or any of its parts for storage or processing [ISO/IEC 2382:2015, Information technology — Vocabulary]

3.1990

input arc (of a transition)

- 1.** arc directed from a place to the transition [ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.1.1]

3.1991

input argument

- 1.** designation given to an operation argument that will always have a value at the invocation of the operation *cf.* output argument

3.1992

input arrow

- 1.** arrow or arrow segment that expresses IDEF0 input [IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.63]

Note 1 to entry: That is, an object type set whose instances are transformed by a function into output. The arrowhead of an input arrow is attached to the left side of a box.

3.1993

input assertion

- 1.** logical expression specifying one or more conditions that program inputs must satisfy in order to be valid *cf.* loop assertion, output assertion, inductive assertion, method

3.1994

input loopback

- 1.** loopback of output from one function to be input for another function in the same diagram [IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.64]

3.1995

input place (of a transition)

- 1.** place connected to the transition by an input arc [ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.20.1]

3.1996

input primitive

- 1.** effort to develop software products, expressed in units of staff-hours

3.1997

input routine

- 1.** those activities required to obtain the logical record to be processed next

Note 1 to entry: If there are no more logical records to be processed, the end-of-input condition becomes true.

3.1998

input-process-output

- 1.** software design technique that consists of identifying the steps involved in each process to be performed and identifying the inputs to and outputs from each step

cf. data structure-centered design, input-process-output chart, modular decomposition, object-oriented design, rapid prototyping

Note 1 to entry: A refinement called hierarchical input-process-output identifies the steps, inputs, and outputs at both general and detailed levels of detail

3.1999

input-process-output (IPO) chart

1. diagram of a software system or module, consisting of a rectangle on the left listing inputs, a rectangle in the center listing processing steps, a rectangle on the right listing outputs, and arrows connecting inputs to processing steps and processing steps to outputs

cf. block diagram, box diagram, bubble chart, flowchart, graph, structure chart

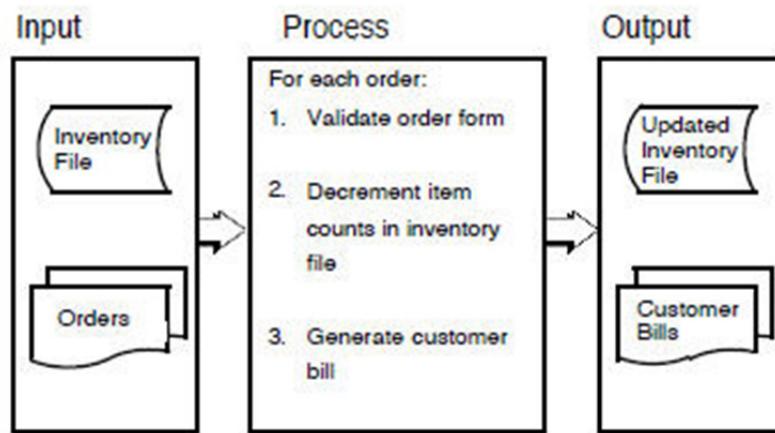


Figure 15 — Input-process-output chart

3.2000

inspection

1. visual examination of a software product to detect and identify software anomalies, including errors and deviations from standards and specifications [*IEEE 1028-2008 IEEE Standard for Software Reviews and Audits*, 3.3]
 2. a static analysis technique that relies on visual examination of development products to detect errors, violations of development standards, and other problems
 3. examining or measuring to verify whether an activity, component, product, result, or service conforms to specified requirements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
 cf. static testing

Note 1 to entry: Inspections are peer examinations led by impartial facilitators who are trained in inspection techniques. Determination of remedial or investigative action for an anomaly is a mandatory element of a software inspection, although the solution could be determined outside the inspection meeting. Types include code inspection and design inspection.

3.2001

inspection-based evaluation

1. evaluation based on the judgment of one or more evaluators who examine or use a system to identify potential usability problems, including deviations from established criteria [*ISO/IEC 25066:2016, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report*, 3.10]

EXAMPLE: heuristic evaluation, cognitive walkthrough, standards inspection, pluralistic walkthrough, consistency inspection

Note 1 to entry: The evaluators making the inspections typically are usability specialists, but can also include end users and members of the design team. Inspection-based evaluation can be conducted by machines in some cases, e.g., when consistency with required terminology is being evaluated. Established criteria typically include user requirements, usability guidelines in standards, design conventions contained in manufacturer guidelines and style guides, task models to be supported, as well as standardized principles.

3.2002

inspections and audits

- 1.** a process to observe performance of contracted work or promised product against agreed to requirements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. audit, inspection

3.2003

installability

- 1.** degree of effectiveness and efficiency with which a product or system can be successfully installed or uninstalled in a specified environment [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.8.2*]

3.2004

installability testing

- 1.** type of portability testing conducted to evaluate whether a test item or set of test items can be installed as required in all specified environments [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.19*]

3.2005

installation and checkout phase

- 1.** period of time in the software life cycle during which a software product is integrated into its operational environment and tested in this environment to ensure that it performs as required

3.2006

installation manual

- 1.** document that provides the information necessary to install a system or component, set initial parameters, and prepare the system or component for operational use
cf. diagnostic manual, operator manual, programmer manual, support manual, user manual

3.2007

installed function point count

- 1.** an application function point count related to a set of installed systems [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

3.2008

instance

- 1.** discrete, bounded thing with an intrinsic, immutable, and unique identity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.89*] **2.** individual occurrence of a type [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*] **3.** mapping of an activity that processes all of its input information and generates all of its output information **4.** of a type, an <X> that satisfies the type [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.21*]

3.2009

instance of benchmarking

- 1.** set of operations, described specifically, used in the execution of a particular benchmarking according to a given method [*ISO/IEC 29155-1:2011 Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions, 2.6*]

3.2010

instance-level attribute

- 1.** mapping from the instances of a class to the instances of a value class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.90*]

3.2011

instance-level operation

- 1.** mapping from the (cross product of the) instances of the class and the instances of the input argument types to the (cross product of the) instances of the other (output) argument types [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.91*]

3.2012

instance-level responsibility

1. responsibility that applies to each instance of the class individually [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.92]
cf. class-level responsibility

3.2013

instantiation

1. process of substituting specific data, instructions, or both into a generic program unit to make it usable in a computer program
2. of an <X> template, an <X> produced from a given <X> template and other necessary information [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.16]
3. identification, for each instance of a life cycle process, of the success criteria, artefact-specific activities and tasks needed to achieve the process outcomes, and the competencies needed to perform these tasks, based on the characteristics and requirements of the target system element [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement*, 3.5]

3.2014

institutional knowledge

1. knowledge from accepted sources, including standards, academic sources, domain and industry bodies of knowledge and organizational knowledge [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement*, 3.4]

3.2015

institutionalization

1. ingrained way of doing business that an organization follows routinely as part of its corporate culture

3.2016

instruction counter

program counter

1. register that indicates the location of the next computer instruction to be executed

3.2017

instruction cycle

1. process of fetching a computer instruction from memory and executing it

cf. instruction time

3.2018

instruction format

1. number and arrangement of fields in a computer instruction

cf. address field, address format, operation field

3.2019

instruction length

1. number of words, bytes, or bits needed to store a computer instruction

cf. instruction format

3.2020

instruction modifier

1. word or part of a word used to alter a computer instruction

3.2021

instruction set

instruction repertoire

1. complete set of instructions recognized by a given computer or provided by a given programming language

3.2022

instruction time

1. time it takes a computer to fetch an instruction from memory and execute it

cf. instruction cycle

3.2023

instructional mode

- 1.** usage mode that is intended to teach the use of software in performing tasks [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.25]

3.2024

instrument

- 1.** in software and system testing, to install or insert devices or instructions into hardware or software to monitor the operation of a system or component

3.2025

instrumentation

- 1.** devices or instructions installed or inserted into hardware or software to monitor the operation of a system or component

3.2026

integer type

- 1.** data type whose members can assume only integer values and can be operated on only by integer arithmetic operations, such as addition, subtraction, and multiplication

cf. character type, enumeration type, logical type, real type

3.2027

integrate

- 1.** to combine software components, hardware components, or both into an overall system **2.** to pull in the changes from one child branch into its parent

3.2028

integrated circuit (IC)

microchip

chip

- 1.** small piece of semiconductive material that contains interconnected electronic elements [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2029

integrated development environment (IDE)

- 1.** set of software tools or applications to provide comprehensive facilities for software development

3.2030

integrated repository

- 1.** planned and controlled storage of information pertinent to the systems engineering effort [*ISO/IEC/IEEE 24748-4:2016, Systems and software engineering-Life cycle management-Part 4: Systems engineering planning*, 4.6]

Note 1 to entry: The integrated repository typically includes key data, e.g., schema, models, tools, technical management decisions, process analysis information, requirement changes, process and product metrics, trade-offs and other analyses.

3.2031

integrated team

- 1.** group of people with complementary skills and expertise who are committed to delivering specified work products in timely collaboration

Note 1 to entry: Integrated team members provide skills and advocacy appropriate to all phases of the work products' life and are collectively responsible for delivering work products as specified. An integrated team includes empowered representatives from organizations, disciplines, and functions that have a stake in the success of the work products.

3.2032

integration

- 1.** process of combining software components, hardware components, or both into an overall system

3.2033

integration test

- 1.** progressive linking and testing of programs or modules in order to ensure their proper functioning in the complete system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. integration testing

3.2034

integration testing

- 1.** testing in which software components, hardware components, or both are combined and tested to evaluate the interaction among them [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.14*]

Note 1 to entry: commonly used for both the integration of components and the integration of entire systems

3.2035

integrity

- 1.** degree to which a system, product, or component prevents unauthorized access to, or modification of, computer programs or data [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.6.2*]
cf. immunity

3.2036

integrity assurance authority

- 1.** independent person or organization responsible for certifying compliance with the integrity-level requirements [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.5.4*]

3.2037

integrity level

- 1.** value representing project-unique characteristic, such as complexity, criticality, risk, safety level, security level, desired performance, and reliability, that define the importance of the system, software, or hardware to the user [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.15*] **2.** degree to which software complies or must comply with a set of stakeholder-selected software and/or software-based system characteristics defined to reflect the importance of the software to its stakeholders **3.** symbolic value representing a degree of compliance within an integrity level scheme **4.** claim of a system, product, or element that includes limitations on a property's values, the claim's scope of applicability, and the allowable uncertainty regarding the claim's achievement [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.3.1*] **5.** required degree of confidence that the system-of-interest meets the associated integrity level claim [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.7*]

Note 1 to entry: Generally, the intention is that maintaining limitations on a property's values related to the relevant items will result in maintaining system risks within limits. The words 'integrity level' form an indivisible label and do not depend on a concept of integrity by itself. An integrity level is different from the likelihood that the integrity level claim is met but they are closely related. The word 'confidence' implies that the definition of integrity levels can be a subjective concept. Integrity levels are defined in terms of risk and hence, cover safety, security, financial and any other dimension of risk that is relevant to the system-of-interest.

3.2038

integrity level assurance authority

- 1.** independent person or organization responsible for certifying compliance with the integrity level requirements [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.8*]

3.2039

integrity level claim

- 1.** claim representing a requirement for a risk reduction measure identified in the risk treatment process of the system-of-interest [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.10*]

3.2040

integrity level definition authority

- 1.** person or organization responsible for defining integrity levels and integrity level requirements [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.9*]

3.2041

integrity level requirement

- 1.** set of requirements that, when met, will provide a level of confidence in the associated integrity level claim commensurate with the associated integrity level [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.11*]

3.2042

integrity level requirements

- 1.** set of specified requirements imposed on aspects related to a system, product, or element and associated activities in order to show the achievement of the assigned integrity level (that is, meeting its claim) within the required limitations on uncertainty; this includes the evidence to be obtained [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.3.2*]

3.2043

integrity level scheme

- 1.** set of system characteristics (such as complexity, risk, safety level, security level, desired performance, reliability, and/or cost) selected as important to stakeholders, and arranged into discrete levels of performance or compliance (integrity levels), to help define the level of quality control to be applied in developing or delivering the software.

3.2044

intellectual property

- 1.** output of creative human thought process that has some intellectual or informational value

EXAMPLE: microcomputer design or computer program

Note 1 to entry: Intellectual property can be protected by patents, copyrights, trademarks, or trade secrets.

3.2045

inter-integrated circuit bus (I2C)

- 1.** bi-directional two-wire serial bus that provides a communication link between integrated circuits

3.2046

interaction

- 1.** action that takes place with the participation of the environment of the object [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.3*] **2.** identity of phenomena existing over some period of time at the interface between two units, caused by one unit and affecting the other unit [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.15*]

Note 1 to entry: The identity is expressed in relevant phenomenological terms. Generally, an interaction identity can be categorized as energy transfer, matter transfer, or information transfer.

3.2047

interaction alias

- 1.** relationship in a build specification that matches an interaction instance in one port with an interaction instance in another port [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.16*]

Note 1 to entry: The interaction alias is used to indicate identity or compatibility of matching interactions at two ports of a port couple or a port alias. In a port alias, both interactions have the same direction. In a port coupling, the interactions have opposite directions. This provides an element of "model bookkeeping" necessary for permitting models developed independently for different units to be integrated into a unified composite model.

3.2048

interaction group

1. subset of the objects participating in a binding managed by the group function [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 13.4.1.1*]

3.2049

interaction point

1. location at which there exists a set of interfaces [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.12*]

3.2050

interactive

1. pertaining to a system or mode of operation in which each user entry causes a response from or action by the system **2.** when the user communicates with the computer in a conversational-type manner [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*] cf. batch, conversational, online, real-time

3.2051

interactive language

1. nonprocedural language in which a program is created as a result of interactive dialog between the user and the computer system

cf. declarative language, rule-based language

Note 1 to entry: The system provides questions, forms, and so on, to aid the user in expressing the results to be achieved.

3.2052

interactive system

1. combination of hardware, software and/or services that receives input from and communicates output to users [*ISO/IEC 25063:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description*]

Note 1 to entry: This includes, where appropriate, packaging, branding, user documentation, online help, support and training.

3.2053

interchange reference point

1. reference point at which an external physical storage medium can be introduced into the system [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 15.3.4*]

3.2054

interconnection

1. association between a computing system tool and something in the environment that affects both endpoints, though not necessarily in the same way [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.7*]

3.2055

interconnection feature

1. property by which members of an interconnection perspective are characterized [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.6*]

3.2056

interconnection group

1. collection of interconnections to a CASE tool that have a common kind of endpoint in the environment [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.7*]

3.2057

interconnection perspective

1. subset of interconnections that share common features in an interconnection group [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.8*]

3.2058

interface

- 1.** shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical signal exchanges, and other characteristics [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **2.** hardware or software component that connects two or more other components for the purpose of passing information from one to the other **3.** to connect two or more components for the purpose of passing information from one to the other **4.** to serve as a connecting or connected component as in (2) **5.** declaration of the meaning and the signature for a property or constraint [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.93] **6.** shared boundary across which information is passed **7.** task's external specification **8.** abstraction of the behavior of an object that consists of a subset of the interactions of that object together with a set of constraints on when they can occur [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 8.5] **9.** description of a set of possible operations that a client is allowed to request of an object [ISO/IEC 19500-1:2012 *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.3.5] **10.** named set of operations that characterize the behavior of an entity [ISO/IEC 19506:2012 *Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.2059

interface control

- 1.** in configuration management, the administrative and technical procedures and documentation necessary to identify functional and physical characteristics between and within configuration items provided by different developers, and to resolve problems concerning the specified interfaces **2.** in configuration management, the process of identifying all functional and physical characteristics relevant to the interfacing of two or more configuration items provided by one or more organizations and ensuring that proposed changes to these characteristics are evaluated and approved prior to implementation
cf. configuration control

3.2060

interface design document (IDD)

- 1.** documentation that describes the architecture and design interfaces between system and components [IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.16] **2.** description of the architecture and design of interfaces between system and components
cf. interface requirements specification (IRS)

Note 1 to entry: These descriptions include control algorithms, protocols, data contents and formats, and performance.

3.2061

interface requirement

- 1.** requirement that specifies an external item with which a system or system component must interact, or that sets forth constraints on formats, timing, or other factors caused by such an interaction
cf. design requirement, functional requirement, implementation requirement, performance requirement, physical requirement

3.2062

interface requirements specification (IRS)

- 1.** documentation that specifies requirements for interfaces between or among systems and components [IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.17]

Note 1 to entry: These requirements include constraints on formats and timing.

3.2063

interface role

- 1.** role of a community, identifying behavior which takes place with the participation of objects that are not members of that community [ISO/IEC 15414:2015 *Information technology — Open distributed processing — Reference model — Enterprise language*, 6.3.5]

3.2064

interface signature

- 1.** set of action signatures associated with the interactions of an interface [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.15]

3.2065

interface specification

- 1.** description of essential functional, performance, and design requirements and constraints at a common boundary between two or more system elements
2. document that specifies the interface characteristics of an existing or planned system or component
cf. interface requirements specification

Note 1 to entry: This includes interfaces between humans and hardware or software, as well as interfaces between humans themselves.

3.2066

interface task

- 1.** task that is part of the application, which interfaces to the external environment

3.2067

interface testing

- 1.** testing conducted to evaluate whether systems or components pass data and control correctly to one another
cf. component testing, integration testing, system testing, unit test

3.2068

interface type

- 1.** type satisfied by any object that satisfies a particular interface [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 3.2.18*]

3.2069

interim function point count

- 1.** a count to determine the size of an interim enhancement during a new development project or an enhancement project [*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*]

Note 1 to entry: i.e., a count to determine the scope of an addition, a change, or a deletion of functional specifications. Both the change in the application function point count and the project function point count can be the subject of this count.

3.2070

interleave

- 1.** to alternate the elements of one sequence with the elements of one or more other sequences so that each sequence retains its identity

EXAMPLE: to alternately perform the steps of two different tasks in order to achieve concurrent operation of the tasks

3.2071

intermediate product

- 1.** system or software product of the development process that is used as inputs to other stages of the development process [*ISO/IEC 25041: 2012 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation guide for developers, acquirers and independent evaluators, 4.10*]

3.2072

intermediate profile

- 1.** profile targeted at very small entities (VSEs) involved in the development of more than one project in parallel with more than one work team [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.30*]

3.2073

intermediate software product needs

- 1.** needs that can be specified as quality requirements by internal measures

3.2074

intermediate system or software product

1. product of the system or software development process that is used as input to another stage of its development process [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.15*]
cf. intermediate product

EXAMPLE: static and dynamic models, other documents and source code

3.2075

intermittent fault

1. temporary or unpredictable fault in a component
cf. random failure, transient error

3.2076

internal action

1. action which takes place without the participation of the environment of the object [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.3*]

3.2077

internal arrow

1. arrow connected at both ends (source and use) to a box in a diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.65*]
cf. boundary arrow

3.2078

internal attribute

1. measurable property of an entity which can be derived purely in terms of the entity itself

Note 1 to entry: Internal attributes are those that relate to the internal organization of the software and its development.

3.2079

internal event

1. means of synchronization between two tasks

3.2080

internal logical file (ILF)

1. user-recognizable group of logically related data or control information maintained within the boundary of the application being measured [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.39*] 2. a logical group of permanent data seen from the perspective of the user that an application uses and maintains [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]
cf. external interface file

Note 1 to entry: The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted. An internal logical file is a type of base functional component.

3.2081

internal measure

1. measure of the product itself, either direct or indirect

Note 1 to entry: The number of lines of code, complexity measures, the number of faults found in a walk through and the Fog Index are all internal measures made on the product itself.

3.2082

internal measure of software quality

1. measure of the degree to which a set of static attributes of a software product satisfies stated and implied needs for the software product to be used under specified conditions [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.16; ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.3.7*]

cf. external measure of software quality

EXAMPLE: Complexity measures and the number, severity, and failure frequency of faults found in a walkthrough are internal software quality measures made on the product itself.

Note 1 to entry: Static attributes include those that relate to the software architecture, structure and its components. Static attributes can be verified by review, inspection, simulation, or automated tools.

3.2083

internal quality

1. totality of attributes of a product that determine its ability to satisfy stated and implied needs when used under specified conditions [*ISO/IEC 14598-1:1999 Information technology — Software product evaluation — Part 1: General overview*, 4.15]

3.2084

internal task-structuring criteria

1. category of the task-structuring criteria addressing how internal objects are mapped to internal tasks and when an internal task is activated

3.2085

internal variability

1. variability defined from an engineer's perspective and not visible to customers [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.6]

3.2086

international standard (IS)

1. standard that is adopted by an international standardizing/standards organization and made available to the public [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.11]

Note 1 to entry: [ISO/IEC Directives, Part 2]

3.2087

internationalization

1. process of developing information so that it is suitable for an international audience [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.26] *cf.* localization

3.2088

Internet

1. worldwide interlinked computer systems and networks connected by gateways that enable the transfer of data between them [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.13]

3.2089

interoperability

1. degree to which two or more systems, products or components can exchange information and use the information that has been exchanged [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.3.2] **2.** ability for two or more ORBs to cooperate to deliver requests to the proper object [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.19] **3.** capability to communicate, execute programs, and transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **4.** capability of objects to collaborate, that is, the capability mutually to communicate information in order to exchange events, proposals, requests, results, commitments and flows [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.5]

cf. compatibility

Note 1 to entry: Interoperability is used in place of compatibility in order to avoid possible ambiguity with replaceability.

3.2090

interoperability testing

1. testing conducted to ensure that a modified system retains the capability of exchanging information with systems of different types, and of using that information

3.2091

interpersonal skills

1. ability to establish and maintain relationships with other people [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2092

interpret

1. to translate and execute each statement or construct of a computer program before translating and executing the next

cf. assemble, compile

3.2093

interpreter

1. computer program that translates and executes each statement or construct of a computer program before translating and executing the next

cf. assembler, compiler

3.2094

interpretive code

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

cf. assembly code, compiler code, machine code

3.2095

interrelationship digraphs

1. a quality management planning tool, the interrelationship digraphs provide a process for creative problem-solving in moderately complex scenarios that possess intertwined logical relationships [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2096

interrogation

1. interaction consisting of one interaction, the invocation, initiated by a client object, resulting in the conveyance of information from that client object to a server object, and requesting a function to be performed by the server object, followed by a second interaction, the termination, initiated by the server object, resulting in the conveyance of information from the server object to the client object in response to the invocation [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.4*]

EXAMPLE: request/response exchange

Note 1 to entry: In interrogations, invocations and terminations are always paired. Announcements do not have terminations. Thus there is no possibility of an operation consisting of an invocation followed by a sequence of associated terminations.

3.2097

Interrupt

interruption

1. suspension of a process to handle an event external to the process **2.** to cause the suspension of a process **3.** loosely, an interrupt request

cf. interrupt latency, interrupt mask, interrupt priority, interrupt service routine, priority interrupt

3.2098

interrupt controller

1. functional unit (integrated circuit) that determines the source and priority of interrupt requests and manages their execution

3.2099

interrupt latency

1. delay between a computer system's receipt of an interrupt request and its handling of the request
cf. interrupt priority

3.2100

interrupt mask

1. mask used to enable or disable interrupts by retaining or suppressing bits that represent interrupt requests

3.2101

interrupt priority

1. importance assigned to a given interrupt request

Note 1 to entry: This importance determines whether the request will cause suspension of the current process and, if there are several outstanding interrupt requests, which will be handled first.

3.2102

interrupt request

1. signal or other input requesting that the currently executing process be suspended to permit performance of another process

3.2103

interrupt service routine

ISR

1. routine that responds to interrupt requests by storing the contents of critical registers, performing the processing required by the interrupt request, restoring the register contents, and restarting the interrupted process

3.2104

interval scale

1. scale in which the measurement values have equal distances corresponding to equal quantities of the attribute
cf. ordinal scale, nominal scale, ratio scale

EXAMPLE: Cyclomatic complexity has the minimum value of one, but each increment represents an additional path. The value of zero is not possible.

3.2105

interviews

1. a formal or informal approach to elicit information from stakeholders by talking to them directly [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2106

interworking reference point

1. reference point at which an interface can be established to allow communication between two or more systems [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 15.3.3]

3.2107

Intranet

1. managed network operating within an organization with controlled and limited access [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.14]

Note 1 to entry: More than one connected or isolated intranet can exist within an organization.

3.2108

intrinsic

1. specification that a property is total (i.e., mandatory), single-valued, and constant [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.94]

3.2109

intrinsic relationship

1. relationship that is total, single-valued, and constant from the perspective of (at least) one of the participating classes, referred to as a dependent class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.95*]
cf. nonintrinsic relationship

EXAMPLE: A transaction has an intrinsic relationship to its related account, because it makes no sense for an instance of a transaction to "switch" to a different account, since that would change the very nature of the transaction.

Note 1 to entry: Such a relationship is considered to be an integral part of the essence of the dependent class.

3.2110

introduction

1. of an <X>, instantiating not achieved by an action of objects in the model [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.19*]
cf. creation

3.2111

invariant

1. assertion that is always true for a specified segment or at a specified point of a computer program **2.** predicate that a specification requires to be true for the entire life time of a set of objects [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.28*]

3.2112

invariant schema

1. set of predicates on one or more information objects which must always be true [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 6.1.1*]

Note 1 to entry: The predicates constrain the possible states and state changes of the objects to which they apply. Thus, an invariant schema is the specification of the types of one or more information objects that will always be satisfied by whatever behavior the objects might exhibit.

3.2113

inverse engineering

1. process of obtaining a high-level representation of the software from the source code
cf. reverse engineering

Note 1 to entry: Inverse engineering provides a more abstract view of the system with the intent of recapturing design and requirements information.

3.2114

invitation for bid (IFB)

1. generally, this term is equivalent to request for proposal. However, in some application areas, it may have a narrower or more specific meaning [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2115

invocation

1. mapping of a parallel initiation of activities of an integral activity group that perform a distinct function and return to the initiating activity.
cf. instance, iteration, mapping

3.2116

invocation deliver

1. signal in the implicitly defined signal interface of a server computational object which has the same name and parameters as the invocation of an interrogation or announcement in the original operation interface [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 3.3.8*]

3.2117

invocation submit

1. signal in the implicitly defined signal interface of a client computational object which has the same name and parameters as the invocation of an interrogation or announcement in the original operation interface [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 3.3.7*]

cf. termination submit

3.2118

IOC

1. initial operational capability [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.2119

IOR

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

3.2120

IOT&E

1. initial operational test and evaluation [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.2121

IP

1. Internet Protocol [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2122

IPO chart

1. input-process-output chart

3.2123

IPR

1. intellectual property rights [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2124

IPSE

1. integrated programming support environment

cf. programming support environment

3.2125

IPT

1. integrated product team [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.2126

IRR

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

3.2127

irreducible

1. decision attribute (criterion) that cannot be expressed in terms of money

3.2128

IRS

1. interface requirements specification.

3.2129

IS

1. International Standard

3.2130

ISO

1. International Organization for Standardization [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 5*]

3.2131

ISO file

1. file image of an entire CD or DVD that is encoded according to ISO 9660 [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]

3.2132

isochronicity

1. relation between adjacent pairs of actions in a sequence, in which every adjacent pair of actions occupies unique, equally-sized, adjacent intervals in time [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.3.2*]

3.2133

ISP

1. in-system programming

3.2134

ISR

2. interrupt service routine

3.2135

issue

1. uniquely identifiable entry in an issue-tracking system that describes a problem or an enhancement **2.** a point or matter in question or in dispute, or a point or matter that is not settled and is under discussion or over which there are opposing views or disagreements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. problem report

Note 1 to entry: The record of an issue includes its identifier and brief description, and often identifies the environment associated with it, its status, severity, priority, and resolution, as well as dependencies, details on replicating or solving a problem, the persons associated with it, attachments, and its change history.

3.2136

issue log

1. a project document used to document and monitor elements under discussion or in dispute between project stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2137

IT

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

3.2138

IT infrastructure

1. all the technical components, system software, databases and data files and deployed application software, technical procedures, and technical documentation used to make the information available [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.22*]

3.2139

IT infrastructure management

1. domain responsible for all of the tasks and activities aimed at managing, maintaining, and renewing the IT infrastructure of the information system, including the operation of the information system [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.23*]

ISO/IEC/IEEE 24765:2017(E)

Note 1 to entry: IT infrastructure management includes all of the tasks, responsibilities and activities that aim for a correct technical operation of the information system, consisting of hardware, (system) software, and data sets. The IT infrastructure management organization is responsible for running the application software in the production environment.

3.2140

IT system

- 1.** system which uses information technologies [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it, 3.3*]

3.2141

iteration

- 1.** process of performing a sequence of steps repeatedly
- 2.** single execution of the sequence of steps in (1)
- 3.** repetition of a process or activity [*ISO/IEC/IEEE 26515: 2011 Systems and software engineering: Developing user documentation in an agile environment, 4.7*]
- 4.** a systematic repetition of one or more software development activities [*Software Extension to the PMBOK® Guide Fifth Edition*]
cf. instance, invocation, mapping

Note 1 to entry: One or more iterations comprise an instance.

3.2142

iterative life cycle

- 1.** a project life cycle where the project scope is generally determined early in the project lifecycle, but time and cost estimates are routinely modified as the project team understanding of the product increases. Iterations develop the product through a series of repeated cycles, while increments successively add to the functionality of the product [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. incremental life cycle

3.2143

IV&V

- 1.** independent verification and validation

3.2144

IVV

- 1.** integration, verification, validation [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 4.2*]

3.2145

IXIT

- 1.** Implementation Extra Information for Testing [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.2146

JCIDS

- 1.** joint capabilities integration and development system [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.2147

JCL

- 1.** job control language

3.2148

JDK

- 1.** Java development kit [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.2149

JFC

- 1.** Java Foundation Class [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2150

JNDI

- 1.** Java naming and directory interface [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.2151

job

- 1.** user-defined unit of work that is to be accomplished by a computer [*ISO/IEC 25023:2016, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality*, 4.3]

cf. job control language, job step, job stream

EXAMPLE: the compilation, loading, and execution of a computer program

3.2152

job control language (JCL)

- 1.** language used to identify a sequence of jobs, describe their requirements to an operating system, and control their execution

3.2153

job function

- 1.** group of engineering processes that is identified as a unit for the purposes of work organization, assignment, or evaluation

EXAMPLE: design, testing, or configuration management

3.2154

job step

- 1.** user-defined portion of a job, explicitly identified by a job control statement

Note 1 to entry: A job consists of one or more job steps.

3.2155

job stream

run stream

- 1.** sequence of programs or jobs set up so that a computer can proceed from one to the next without the need for operator intervention

3.2156

join

- 1.** junction at which an arrow segment (going from source to use) merges with one or more other arrow segments to form a root arrow segment [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.66]

Note 1 to entry: can denote bundling of arrows, meaning the inclusion of multiple object types within an object type set

3.2157

joining action

- 1.** action shared between two or more chains resulting in a single chain [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.1.3]

3.2158

JPEG

- 1.** Joint Photographic Experts Group [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

Note 1 to entry: image format

3.2159

JPG

- 1.** Joint Photographic Group [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

Note 1 to entry: image format

3.2160

JTA

- 1.** Java Transaction API [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.2161

jump

- 1.** to depart from the implicit or declared order in which computer program statements are being executed **2.** program statement that causes a departure as in (1) **3.** departure described in (1)
cf. transfer

3.2162

junction

- 1.** point at which either a root arrow segment divides into branching arrow segments or arrow segments join into a root arrow segment [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.67*]

3.2163

KDM

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

3.2164

KDM entity

- 1.** meta-model element (as well as the corresponding model elements) that represents a thing of significance of the system of interest, about which information needs to be known or held [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.2165

KDM instance

- 1.** collection of KDM model elements that represent one or more views of the system of interest [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.2166

KDM model

- 1.** meta-model element (as well as the corresponding model elements) that is a container for a KDM view [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.2167

KDM relationship

- 1.** meta-model element (as well as the corresponding model elements) that represents some semantic association between elements of the system of interest [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.2168

kernel

resident control program

- 1.** that portion of an operating system that is kept in main memory at all times **2.** a software module that encapsulates an elementary function or functions of a system
cf. nucleus

3.2169

kernel entity

- 1.** classification used for a meta-entity whose instances can exist without the occurrences of other meta-entities
[*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

EXAMPLE: An instance of the meta-entity called Attribute, having a name, full description and brief description, is significant without the knowledge of the DataObject it describes.

3.2170

key migration

- 1.** the modeling process of placing the primary key of a parent or generic entity in its child or category entity as a foreign key [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.96]

Note 1 to entry: [key style]

3.2171

key-style view

- 1.** view that represents the structure and semantics of data within an enterprise [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.97]

Note 1 to entry: That is, data (information) models.

3.2172

knowledge

- 1.** aspect of an instance's specification that is determined by the values of its attributes, participant properties, and constant, read-only operations [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.98]

3.2173

knowledge base (K-base)

- 1.** database that contains inference rules and information about human experience and expertise in a domain [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: In self-improving systems, the knowledge base additionally contains information resulting from the solution of previously encountered problems

3.2174

known error

- 1.** result of a problem with an identified root cause or an identified workaround that reduces or eliminates its impact

3.2175

KOPS

- 1.** kilo-operations per second; that is, thousands of operations per second
cf. MFLOPS, MIPS

Note 1 to entry: a measure of computer processing speed

3.2176

KPP

- 1.** key performance parameter [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2177

KSA

- 1.** key system attribute [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2178

label

- 1.** name or identifier assigned to a computer program statement to enable other statements to refer to that statement **2.** one or more characters, within or attached to a set of data, that identify or describe the data **3.** word or phrase that is attached to, or part of, a model graphic [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.99*] **4.** information associated with the net graph or one of its objects [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.5*]

3.2179

lag

- 1.** the amount of time whereby a successor activity is required to be delayed with respect to a predecessor activity [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2180

language

- 1.** systematic means of communicating ideas by the use of conventionalized signs, sounds, gestures, or marks and rules for the formation of admissible expressions **2.** means of communication, with syntax and semantics, consisting of a set of representations, conventions, and associated rules used to convey information

3.2181

language binding

language mapping

- 1.** means and conventions by which a programmer writing in a specific programming language accesses ORB capabilities [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 3.2.20*]

3.2182

language processor

- 1.** computer program that translates, interprets, or performs other tasks required to process statements expressed in a given language

cf. assembler, compiler, interpreter, translator

3.2183

language standard

- 1.** standard that describes the characteristics of a language used to describe a requirements specification, a design, or test data

3.2184

late binding

- 1.** the assignment of tasks to specific resources when the resources are available to start work, rather than when the project is planned [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.2185

late finish date (LF)

- 1.** in the Critical Path Method, the latest possible point in time when the uncompleted portions of a schedule activity can finish based on the schedule network logic, the project completion date, and any schedule constraints [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2186

late start date (LS)

- 1.** in the Critical Path Method, the latest possible point in time when the uncompleted portions of a schedule activity can start based on the schedule network logic, the project completion date, and any schedule constraints [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2187

latency

- 1.** time interval between the instant at which an instruction control unit issues a call for data and the instant at which the transfer of data is started

3.2188

latent variable

- 1.** variable representing a unidimensional construct [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.8*]

3.2189

lateral compression

- 1.** in software design, a form of demodularization in which two or more modules that execute one after the other are combined into a single module
cf. downward compression, upward compression

3.2190

layer

- 1.** partition resulting from the functional division of a software system, where layers are organized in a hierarchy; there is only one layer at each level in the hierarchy; there is a superior/subordinate hierarchical dependency between the functional services provided by software in any two layers in the software system that exchange data directly; and the software in any two layers in the software system that exchange data interpret only part of that data identically [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.15*]

3.2191

layout

- 1.** physical organization of source code including the use of white space, grouping, blank lines, alignment, indentation, and parentheses

3.2192

layout chart

- 1.** sheet provided with scales and other indicators conforming to the characteristics of the majority of character printing machines in general office and data processing use [*ISO 3535:1977 Forms design sheet and layout chart, 4.2*]

3.2193

LCC

- 1.** life-cycle cost [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.2194

LCCE

- 1.** life-cycle cost estimate [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.2195

LCSP

- 1.** life cycle sustainment plan [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.2196

lead

- 1.** the amount of time whereby a successor activity can be advanced with respect to a predecessor activity [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. lag

3.2197

leadframe

- 1.** framework of a semiconductor device, made of plated metal

3.2198

leading decision

- 1.** loop control that is executed before the loop body

cf. trailing decision, WHILE

3.2199

leaf diagram

- 1.** diagram that has no descendent diagrams [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.68]

Note 1 to entry: That is, a diagram that does not contain any function that has been decomposed.

3.2200

leaf node

- 1.** function that is not decomposed [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.69]

Note 1 to entry: A box that represents a leaf node does not have a box detail reference.

3.2201

learnability

- 1.** degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.4.2]

Note 1 to entry: Can be specified or measured either as the extent to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use, or by product properties corresponding to suitability for learning as defined in ISO 9241-110.

3.2202

legacy software

- 1.** software originally created without information structures [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.17]

3.2203

lessons learned

- 1.** [Output/Input] the learning gained from the process of performing the project. Lessons learned may be identified at any point. Also considered a project record, to be included in the lessons learned knowledge base. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** the knowledge gained during a project which shows how project events were addressed or should be addressed in the future with the purpose of improving future performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2204

lessons learned knowledge base

- 1.** a store of historical information and lessons learned about both the outcomes of previous project selection decisions and previous project performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2205

level

- 1.** designation of the coverage and detail of a view [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.100]

Note 1 to entry: There are multiple levels of view; each is intended to be distinct, specified in terms of the modeling constructs to be used.

3.2206

level of abstraction

- 1.** view of an object at a specific level of detail [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.12]

3.2207

level of effort (LOE)

- 1.** an activity that does not produce definitive end products and is measured by the passage of time [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

EXAMPLE: customer liaison, project cost accounting, project management

Note 1 to entry: One of three EVM types of activities used to measure work performance

3.2208

level of risk

- 1.** magnitude of a risk or combination of risks, expressed in terms of the combination of consequences and their likelihood [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.12]

3.2209

lexicography

- 1.** decision technique that prioritizes the decision attributes
cf. dominance, satisficing

3.2210

LF

- 1.** late finish date [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2211

LFT&E

- 1.** live fire test and evaluation [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2212

liaison

- 1.** relationship between a set of objects which results from the performance of some establishing behavior [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.2.4] **2.** state of having a contractual context in common [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.2.4]

3.2213

license

- 1.** legal agreement between two parties, the licensor and the licensee, as to the terms and conditions for the use or transfer of an intellectual property right from the licensor to the licensee

3.2214

license compliance audit

- 1.** audit that reconciles license-related information from multiple information sources, such as entitlement consumption against entitlement rights

3.2215

license model

- 1.** class of licenses with common characteristics

EXAMPLE: site license, OEM license, per computer license

3.2216

licensing standard

- 1.** standard that describes the characteristics of an authorization given by an official or a legal authority to an individual or organization to do or own a specific thing

3.2217

life cycle

- 1.** evolution of a system, product, service, project or other human-made entity from conception through retirement [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.16; *ISO/IEC TS*

24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.24; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.23J

3.2218

life cycle cost

- 1.** total investment in product development, manufacturing, test, distribution, operation, support, training, and disposal

3.2219

life cycle model

- 1.** framework of processes and activities concerned with the life cycle that may be organized into stages, which also acts as a common reference for communication and understanding [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.24*] **2.** framework containing the processes, activities, and tasks involved in the development, operation, and maintenance of a software product, spanning the life of the system from the definition of its requirements to the termination of its use [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.1*]

3.2220

life cycle processes

- 1.** set of interrelated activities that result in the development or assessment of system, software, or hardware products [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*;

Note 1 to entry: Each activity consists of tasks. The life cycle processes can overlap one another. For V&V purposes, no process is concluded until its development products are verified and validated according to the defined tasks in the validation and verification plan.

3.2221

lifeline

- 1.** any historical sequence of behaviors through which a unit is manipulated by external stimuli [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.17*]

3.2222

lightweight process

- 1.** process with a single thread of control; a task

3.2223

likelihood

- 1.** probability of something happening [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.13*]

Note 1 to entry: Quantitative expressions include numerical scales or probabilities.

3.2224

limit

- 1.** restriction on rights or privileges granted by a software entitlement [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.16*]

3.2225

limited entry table

- 1.** decision table where all the conditions and actions are completely described without reference to the rules [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.14*]

3.2226

line of code

- 1.** programming-language statement; a non-comment, nonblank deliverable source statement

3.2227

linear interpolation

- 1.** approximation of the value of a function at a given point, based on values on a straight line between two known points

3.2228

link

hyperlink

- 1.** to create a load module from two or more independently translated object modules or load modules by resolving cross-references among them **2.** part of a computer program, often a single instruction or address, which passes control and parameters between separate modules of the program **3.** to provide a link as in (2) **4.** navigation method that takes the user from one item of on-screen documentation to another item [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.21] **5.** reference from some part of one document to some part of another document or another part of the same document [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.15]

cf. linkage editor

3.2229

linkage editor

linker

- 1.** computer program that creates a single load module from two or more independently translated object modules or load modules by resolving cross-references among the modules and, possibly, by relocating elements
cf. linking loader

Note 1 to entry: can be part of a loader

3.2230

linking loader

- 1.** computer program that reads one or more object modules into main memory in preparation for execution, creates a single load module by resolving cross-references among the separate modules, and, in some cases, adjusts the addresses to reflect the storage locations into which the code has been loaded
cf. absolute loader, relocating loader, linkage editor

3.2231

list

- 1.** set of data items, each of which has the same data definition **2.** to print or otherwise display a set of data items
3. collection class that contains no duplicates and whose members are ordered [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.101]

3.2232

list function

- 1.** online function that displays an overview of entity type occurrences that possibly satisfy a certain selection criterion [*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*]

3.2233

list processing language

- 1.** programming language designed to facilitate the manipulation of data expressed in the form of lists
cf. algebraic language, algorithmic language, logic programming language

EXAMPLE: LISP, IPL

3.2234

listing

- 1.** ordered display or printout of data items, program statements, or other information

3.2235

literal

- 1.** in a source program, an explicit representation of the value of an item **2.** denotation of a specific instance of a value class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.102] **3.** number or string that is used by a program directly rather than being embedded in a named constant or variable

3.2236

LITSR

1. level interim test status report

3.2237

load

1. to read machine code into main memory in preparation for execution and, in some cases, to perform address adjustment and linking of modules **2.** to copy computer instructions or data from external storage to internal storage or from internal storage to registers

cf. loader

3.2238

load map

1. computer-generated list that identifies the location or size of all or selected parts of memory-resident code or data

3.2239

load module

1. computer program or subprogram in a form suitable for loading into main storage for execution by a computer; usually the output of a linkage editor

cf. object module

3.2240

load testing

1. type of performance efficiency testing conducted to evaluate the behavior of a test item under anticipated conditions of varying load, usually between anticipated conditions of low, typical, and peak usage [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.20]

3.2241

load-and-go

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

3.2242

loaded origin

1. address of the initial storage location of a computer program at the time the program is loaded into main memory

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

3.2243

loader

1. computer program that reads machine code into main memory in preparation for execution and, in some cases, adjusts the addresses and links the modules **2.** any program that reads programs or data into main memory

cf. bootstrap, linkage editor

Note 1 to entry: Types include absolute loader, linking loader, relocating loader.

3.2244

local area network (LAN)

1. computer network located on a user's premises within a limited geographical area [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

Note 1 to entry: Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary can be subject to some form of regulation.

3.2245

local compaction

1. in microprogramming, compaction in which microoperations are not moved beyond the boundaries of the single-entry, single-exit sequential blocks in which they occur

cf. global compaction

3.2246

local customization

1. FSM method that has been modified for local use, such that it might produce different functional sizes from those obtained prior to modification [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.9*]

3.2247

local data

1. data that can be accessed by only one module or set of nested modules in a computer program **2.** data that can be accessed only within the routine in which it is declared

cf. global data

3.2248

local SAM owner

1. individual at a level of the organization below that of the SAM owner who is identified as being responsible for SAM for a defined part of the organization [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.21*]

3.2249

local variable

1. variable that can be accessed by only one module or set of nested modules in a computer program

cf. global variable

3.2250

locality of quality responsibility

1. assignment of responsibility for specific quality-related requirements or decompositions thereof to a particular process instance [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.6*] **2.** responsibility for achievement of a quality requirement or decomposition thereof [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.8*]

3.2251

localization

1. creation of a national or specific regional version of a product [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.28*]

cf. internationalization

Note 1 to entry: Localization can be carried out separately from the translation process.

3.2252

location facility

1. set of service primitives that allow a client-side binder object to ask a server-side if it will accept requests carrying invocations to a particular (computational) server object [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 3.3.9*]

Note 1 to entry: The server-side can confirm or reject the proposal or suggest an alternative server-side that is capable of handling requests.

3.2253

location in space

1. interval of arbitrary size in space at which an action can occur [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.10*]

3.2254

location in time

1. interval of arbitrary size in time at which an action can occur [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.11*]

3.2255

location reference

- 1.** indicator following a heading or subheading in an index or table of contents, showing to which part of the document the heading or subheading refers

3.2256

location transparency

- 1.** distribution transparency which masks the use of information about location in space when identifying and binding to interfaces [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.4.1.3]

3.2257

lock

- 1.** exclusive permission to edit a file

3.2258

lockout

- 1.** computer resource allocation technique in which shared resources (especially data) are protected by permitting access by only one device or process at a time

cf. deadlock, semaphore

3.2259

LOE

- 1.** level of effort [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2260

log

- 1.** a document used to record and describe or denote selected items identified during execution of a process or activity. Usually used with a modifier, such as issue, quality control, action, or defect [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2261

log off

log out

- 1.** to end a session [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2262

log on

log in

- 1.** to initiate a session [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2263

logic

- 1.** logical restrictions (conditions) on actions [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.2264

logic programming language

- 1.** a programming language used to express programs in terms of control constructs and a restricted predicate calculus

cf. algebraic language, algorithmic language, list processing language

EXAMPLE: PROLOG

3.2265

logical cohesion

- 1.** type of cohesion in which the tasks performed by a software module perform logically similar functions
cf. coincidental cohesion, communicational cohesion, functional cohesion, procedural cohesion, sequential cohesion, temporal cohesion

EXAMPLE: processing of different types of input data

3.2266

logical file

- 1.** a logical group of permanent data seen from the perspective of the user [*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*]
cf. data function

Note 1 to entry: It is an internal logical file or an external interface file.

3.2267

logical layout

- 1.** the set of user required data element types and their logical structure as defined for an output product, apart from aspects of physical implementation [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

Note 1 to entry: This does not pertain to the physical way in which data is presented on a screen, report, or other media.

3.2268

logical record

- 1.** set of data which is processed in a single iteration of the main procedure

EXAMPLE: a row in a database table

Note 1 to entry: It can be part or the whole of a single physical record or of a number of records.

3.2269

logical relationship

dependency

- 1.** a dependency between two activities, or between an activity and a milestone [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. precedence relationship

3.2270

logical source statement (LSS)

- 1.** software instruction, independent of the physical format (lines of code) in which it appears
cf. physical source statement

3.2271

logical trace

- 1.** execution trace that records only branch or jump instructions
cf. execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

3.2272

logical transaction

- 1.** the basic functional component of Mk II FPA [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

Note 1 to entry: The smallest complete unit of information processing that is meaningful to the end user in the business. It is triggered by an event in the real world of interest to the user, or by a request for information. It comprises an input, process and output component. It must be self-contained and leave the application being counted in a consistent state.

3.2273

logical type

- 1.** data type whose members can assume only logical values (usually TRUE and FALSE) and can be operated on only by logical operators, such as and, OR, and NOT
cf. character type, enumeration type, integer type, real type

3.2274

loop

iterative construct

- 1.** sequence of computer program statements that is executed repeatedly until a given condition is met or while a given condition is true **2.** to execute a sequence of computer program statements as in (1)
cf. loop body, loop control, UNTIL, WHILE

3.2275

loop assertion

loop invariant

- 1.** logical expression specifying one or more conditions that must be met each time a particular point in a program loop is executed

cf. input assertion, output assertion, inductive assertion method

3.2276

loop body

- 1.** part of a loop that accomplishes the loop's primary purpose
cf. loop control

3.2277

loop control

- 1.** part of a loop that determines whether to exit from the loop
cf. loop body, leading decision, trailing decision

3.2278

loop-control variable

- 1.** program variable used to determine whether to exit from a loop

3.2279

loopback

- 1.** an internal arrow that is the output of a box whose box number is greater than the box number of the box that uses that arrow as input, control, or mechanism [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.70]

Note 1 to entry: These uses are referred to as input loopback, control loopback, and mechanism loopback, respectively.

3.2280

loopback testing

- 1.** testing in which signals or data from a test device are input to a system or component, and results are returned to the test device for measurement or comparison

3.2281

low level

- 1.** specific; detailed

3.2282

low-level design

- 1.** process of design at the individual-routine or, sometimes, class level under the guidance of a more general design

cf. detailed design

3.2283

low-profile quad flat package (LQFP)

- 1.** semiconductor device based on a leadframe with gull wing-shaped leads on all four sides

Note 1 to entry: The LQFP package can be encapsulated in plastic.

3.2284

lowclass

1. if an instance is in a class S and not in any subclass of S, then S is the lowclass for the instance [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.103*]

3.2285

LQFP

1. low-profile quad flat package

3.2286

LRIP

1. low-rate initial production [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.2287

LS

1. late start date [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2288

LTC

1. level test case.

3.2289

LTD

1. level test design.

3.2290

LTL

1. level test log.

3.2291

LTP

1. level test plan.

cf. LTPr

3.2292

LTPr

1. level test procedure.]

cf. LTP

3.2293

LTR

1. level test report.

3.2294

M&S

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

3.2295

MAC

1. Media Access Control [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2296

machine code

1. computer instructions and data definitions expressed in a form that can be recognized by the processing unit of a computer

cf. assembly code, compiler code, interpretive code

3.2297

machine language

first-generation language
machine-oriented language

1. language that can be recognized by the processing unit of a computer

cf. assembly language, fifth-generation language, fourth-generation language, high-order language, symbolic language

Note 1 to entry: Such a language usually consists of patterns of 1s and 0s, with no symbolic naming of operations or addresses.

3.2298

machine-dependent

1. pertaining to software that relies on features unique to a particular type of computer and therefore executes only on computers of that type

cf. machine-independent

3.2299

machine-independent

1. pertaining to software that does not rely on features unique to a particular type of computer, and therefore executes on computers of more than one type

cf. machine-dependent, portability

3.2300

machine-readable

1. pertaining to data in a form that can be automatically input to a computer

EXAMPLE: data encoded on a diskette

3.2301

macro

macro definition

1. in software engineering, a predefined sequence of computer instructions that is inserted into a program, usually during assembly or compilation, at each place that its corresponding macroinstruction appears in the program
cf. macroinstruction, macrogenerator, open subroutine

3.2302

macro library

1. collection of macros available for use by a macrogenerator

cf. system library

3.2303

macroassembler

1. assembler that includes, or performs the functions of, a macrogenerator

3.2304

macrogenerator

macro-generating program

1. routine, often part of an assembler or compiler, that replaces each macroinstruction in a source program with the predefined sequence of instructions that the macroinstruction represents

3.2305

macroinstruction

1. source code instruction that is replaced by a predefined sequence of source instructions, usually in the same language as the rest of the program and usually during assembly or compilation

cf. macro, macrogenerator

3.2306

macroprocessor

1. routine or set of routines provided in some assemblers and compilers to support the definition and use of macros

3.2307

macroprogramming

1. computer programming using macros and macroinstructions

3.2308

magic number

magic string

1. literal value that is used by a program directly rather than being embedded in a named constant or variable
cf. literal

3.2309

magnetic core memory

1. volatile memory that uses magnetic rings as the storage element

3.2310

main procedure

1. all those activities subsequent to the general initiation routine and prior to the general termination routine within the complete procedure

3.2311

main program

1. software component that is called by the operating system of a computer and that usually calls other software components

cf. routine, subprogram

3.2312

mainframe

1. computer intended to run in a computer center, with extensive capabilities and resources to which other computers can be connected so that they can share facilities [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.2313

maintain

1. add, change or delete data through an elementary process [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.40*]

3.2314

maintainability

1. ease with which a software system or component can be modified to change or add capabilities, correct faults or defects, improve performance or other attributes, or adapt to a changed environment 2. ease with which a hardware system or component can be retained in, or restored to, a state in which it can perform its required functions 3. capability of the software product to be modified [IEEE 14764-2006 *Software Engineering - Software Life Cycle Processes - Maintenance, 3.4*] 4. average effort required to locate and fix a software failure 5. speed and ease with which a program can be corrected or changed [IEEE 982.1-2005 *IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, 2.3*] 6. degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.7*]
cf. extendability, flexibility

Note 1 to entry: Maintainability includes installation of updates and upgrades. Modifications include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications. Modifications include those carried out by specialized support staff, and those carried out by business or operational staff, or end users.

3.2315

maintainability plan

1. document setting out the specific maintainability practices, resources and sequence of activities relevant to software

Note 1 to entry: The developer prepares the Maintainability Plan.

3.2316

maintainability testing

- 1.** test type conducted to evaluate the degree of effectiveness and efficiency with which a test item can be modified
[*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.21]

3.2317

maintainer

- 1.** individual or organization that performs maintenance activities [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.17] **2.** organization that performs maintenance activities [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.18]

3.2318

maintenance

- 1.** process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.9] **2.** process of retaining a hardware system or component in, or restoring it to, a state in which it can perform its required functions

cf. adaptive maintenance, corrective maintenance, perfective maintenance, software maintenance

3.2319

maintenance branch

- 1.** branch where most development concerns bug fixes

3.2320

maintenance enhancement

- 1.** modification to an existing software product to satisfy a new requirement [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance*, 3.5]

Note 1 to entry: There are two types of software enhancements, adaptive and perfective. A maintenance enhancement is not a software correction.

3.2321

maintenance manual (MM)

- 1.** software engineering project-deliverable document that enables a system's maintenance personnel (rather than users) to maintain the system

cf. operator manual, user manual

3.2322

maintenance personnel

- 1.** software engineers who maintain software systems

3.2323

maintenance plan

- 1.** document setting out the specific maintenance practices, resources, and sequence of activities relevant to maintaining a software product

3.2324

maintenance program

maintenance infrastructure

- 1.** organizational structure, responsibilities, procedures, processes, and resources used for implementing the maintenance plan

3.2325

maintenance project

- 1.** software development project described as maintenance to correct errors in an original requirements specification, to adapt a system to a new environment, or to enhance a system

3.2326

majority

1. support from more than 50 percent of the members of the group [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2327

make or buy analysis

1. the process of gathering and organizing data about product requirements and analyzing them against available alternatives including the purchase or internal manufacture of the product [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2328

make or buy decision

1. decisions made regarding the external purchase or internal manufacture of a product [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2329

manage

1. [requirements] provide storing and editing capabilities, tracking history of edition, versioning, author identification, change management, time stamping, user notification for content changes, security rights control [*ISO/IEC TR 24766:2009 Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities, 3.2*]

3.2330

manage communications

1. the process of creating, collecting, distributing, storing, retrieving and the ultimate disposition of project information in accordance to the Communications Management Plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2331

manage project team

1. the process of tracking team member performance, providing feedback, resolving issues, and managing team changes to optimize project performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2332

manage stakeholder engagement

1. the process of communicating and working with stakeholders to meet their needs / expectations, address issues as they occur, and foster appropriate stakeholder engagement in project activities throughout the project life cycle [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2333

manageability

1. degree to which IT infrastructure management can attain and keep an application in its operational state [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.24*]

Note 1 to entry: Manageability involves the transparency and manageability of applications from an infrastructure point of view.

3.2334

managed network

1. network or set of networks established and controlled by one or more organizations to meet specific organizational or business needs [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.16*]

3.2335

managed process

1. performed process that is planned and executed in accordance with policy; employs skilled people having adequate resources to produce controlled outputs; involves relevant stakeholders; is monitored, controlled, and reviewed; and is evaluated for adherence to its process description

cf. performed process

3.2336

managed role

- 1.** view of the management interface of an object which is being managed within an ODP system [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 14.4]

3.2337

managed website

- 1.** site created and maintained based on organizational guidelines [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.17]

3.2338

management

- 1.** system of controls and processes required to achieve the strategic objectives set by the organization's governing body

Note 1 to entry: Management is subject to the policy guidance and monitoring set through corporate governance.

3.2339

management information

- 1.** knowledge concerning objects which are of relevance to management [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 14.3]

3.2340

management notification

- 1.** event notification initiated by an object operating in a managed role [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 14.6]

3.2341

management process

- 1.** activities that are undertaken in order to ensure that the software engineering processes are performed in a manner consistent with the organization's policies, goals, and standards.

3.2342

management reserve

- 1.** an amount of the project budget withheld for management control purposes. These are budgets reserved for unforeseen work that is within scope of the project. The management reserve is not included in the Performance Measurement Baseline (PMB) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2343

management review

- 1.** systematic evaluation of a software acquisition, supply, development, operation, or maintenance process performed by or on behalf of management that monitors progress, determines the status of plans and schedules, confirms requirements and their system allocation, or evaluates the effectiveness of management approaches used to achieve fitness for purpose [*IEEE 1028-2008 IEEE Standard for Software Reviews and Audits*, 3.4]

3.2344

management skills

- 1.** the ability to plan, organize, direct, and control individuals or groups of people to achieve specific goals [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2345

management system

- 1.** set of interrelated or interacting elements to establish policy and objectives and to achieve those objectives

Note 1 to entry: [ISO 9000:2005]

3.2346

managerial independence

1. of software quality assurance (SQA), situation in which the responsibility of the SQA effort is vested in an organization separate from the development and project management organizations [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2]

3.2347

managing role

1. view of an object which is performing managing actions [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 14.5]

3.2348

mandatory

1. syntax keyword used to specify a total mapping [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.104]
cf. optional, total

3.2349

mandatory dependency

hard logic

1. a relationship that is contractually required or inherent in the nature of the work [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2350

mandatory element

1. element that is required to be present in a tag in order to claim conformance with a standard

3.2351

mandatory nonidentifying relationship

1. nonidentifying relationship in which an instance of the child entity must be related to an instance of the parent entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.105]

cf. optional nonidentifying relationship, nonidentifying relationship [key style]

3.2352

manufacture

1. in software engineering, the process of copying software to disks, chips, or other devices for distribution to customers or users

3.2353

manufacturing phase

1. period of time in the software life cycle during which the basic version of a software product is adapted to a specified set of operational environments and is distributed to a customer base

3.2354

many-sorted algebra

1. mathematical structure comprising a set of sets and a set of functions taking these sets as domains and co-domains [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.13]

3.2355

many-to-many relationship

1. relationship between two state classes (not necessarily distinct) in which each instance of one class can be associated with any number of instances of a second class (possibly none), and each instance of the second class can be related to any number of instances of the first class (possibly none) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.106]

3.2356

map program

1. software tool, often part of a compiler or assembler, that generates a load map

3.2357

mapping

- 1.** assigned correspondence between two things that is represented as a set of ordered pairs [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.107*] **2.** establishing a sequence of activities according to a selected software life cycle model (SLCM).
cf. instance, invocation, iteration, software life cycle model (SLCM)

3.2358

mapping completeness

- 1.** a designation of whether a mapping is complete (totally mapped) or incomplete (partial) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.108*]
cf. partial, total

3.2359

market research

- 1.** the process of gathering information at conferences, online reviews and a variety of sources to identify market capabilities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2360

marking (of a net)

- 1.** the set of the place markings for all places of the net [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.14*]

3.2361

marking of a place

- 1.** a multiset of tokens associated with ('residing in') the place [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.14.3*]

3.2362

MARR

- 1.** minimum attractive rate of return

3.2363

mask

- 1.** a pattern of bits or characters designed to be logically combined with an unknown data item to retain or suppress portions of the data item
cf. interrupt mask

EXAMPLE: The bit string '00000011' when logically ANDed with an eight-bit data item, gives a result that retains the last two bits of the data item and has zero in all the other bit positions.

3.2364

mask ROM

- 1.** read-only memory unit whose circuits are programmed during the manufacturing process

3.2365

master data

- 1.** data held by an organization that describes the entities that are both independent and fundamental for an enterprise that it needs to reference in order to perform its transaction [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 4.25*]

3.2366

master library

- 1.** a software library containing master copies of software and documentation from which working copies can be made for distribution and use
cf. production library, software development library, software repository, system library

3.2367

master schedule

1. a summary-level project schedule that identifies the major deliverables and work breakdown structure components and key schedule milestones [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] *cf.* milestone schedule

3.2368

material

materiel

1. the aggregate of things used by an organization in any undertaking, such as equipment, apparatus, tools, machinery, gear, material, and supplies [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2369

matrix diagrams

1. a quality management and control tool used to perform data analysis within the organizational structure created in the matrix. The matrix diagram seeks to show the strength of relationships between factors, causes and objectives that exist between the rows and columns that form the matrix. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2370

matrix organization

1. any organizational structure in which the project manager shares responsibility with the functional managers for assigning priorities and for directing the work of persons assigned to the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2371

maturity

1. degree to which a system, product or component meets needs for reliability under normal operation [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.5.1*]

Note 1 to entry: The concept of maturity can be applied to quality characteristics to indicate the degree to which they meet required needs under normal operation.

3.2372

maturity level

1. point on an ordinal scale of organizational process maturity that characterizes the maturity of the organizational unit assessed in the scope of the maturity model used [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.4.1*]

3.2373

maturity model

1. model derived from one or more specified process assessment model(s) that identifies the process sets associated with the levels in a specified scale of organizational process maturity [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.3.7*]

3.2374

maximax rule

1. in decision making under uncertainty, assuming that the best state of nature will happen, selection of the alternative that has the best payoff from all of the best payoffs

cf. Hurwicz criterion, maximin rule, minimax regret rule

3.2375

maximin rule

1. in decision making under uncertainty, assuming that the worst state of nature will happen, selection of the alternative that has the best payoff from all of the worst payoffs

cf. Hurwicz criterion, maximax rule, minimax regret rule

Note 1 to entry: the most pessimistic of the uncertainty techniques

3.2376

MBLa

- 1.** manage benchmarking business level activity [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking, 4*]

3.2377

MCU

- 1.** microcontroller unit

3.2378

MD5

- 1.** message digest 5 [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.22*]

3.2379

MDA

- 1.** Model Driven Architecture ® [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, A.8*]

3.2380

mean execution time

- 1.** the mean value of all execution times of tasks of the j-the task type which were submitted within the rating interval [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.8*]

3.2381

mean execution time rating value

- 1.** the quotient (corresponding to the j-th task type) of the mean execution time reference value and the measured mean execution time [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.9*]

3.2382

mean execution time reference value

- 1.** the mean execution time maximally accepted by the emulated user [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.1*]

3.2383

mean time between failures (MTBF)

- 1.** the expected or observed time between consecutive failures in a system or component
cf. up time

3.2384

mean time to repair (MTTR)

- 1.** expected or observed duration required to return a malfunctioning system or component to normal operations
2. the mean time the maintenance team requires to implement a change and restore the system to working order
cf. down time

3.2385

meaning (of a responsibility)

- 1.** statement of what the responsibility means [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.109*]

Note 1 to entry: The statement of responsibility is written from the point of view of the requester, not the implementer. The statement of responsibility states what the requester needs to know to make intelligent use of the property or constraint. That statement is complete enough to let a requester decide whether to make the request, but it stops short of explaining how a behavior or value is accomplished or derived. Meaning is initially captured using freeform natural language text in a glossary definition. It can be more formally refined into a statement of pre-conditions and post-conditions using the specification language.

3.2386

meaningful

1. user-recognizable and satisfies a functional user requirement [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.41]

3.2387

measurable concept

1. abstract relationship between attributes of entities and information needs [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.14]

3.2388

measurand

1. particular quantity subject to measurement [ISO/IEC TR 14143-3:2003 *Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*, 3.5]

EXAMPLE: vapor pressure of a given sample of water at 20 °C

Note 1 to entry: The specification of a measurand involves statements about quantities such as time, temperature and pressure.

3.2389

measure

1. variable to which a value is assigned as the result of measurement [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.15; ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.18] **2.** make a measurement [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.16; ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.19] **3.** way to ascertain or appraise value by comparing it to a norm [IEEE 1061-1998 (R2004) *IEEE Standard for Software Quality Metrics Methodology*, 2.6] **4.** to apply a metric [IEEE 1061-1998 (R2004) *IEEE Standard for Software Quality Metrics Methodology*, 2.6] **5.** act or process of measuring [IEEE 982.1-2005 *IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*, 2.4] **6.** number or symbol assigned to an entity by a mapping from the empirical world to the formal, relational world in order to characterize an attribute [IEEE 982.1-2005 *IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*, 2.4]

3.2390

measure of effectiveness (MOE)

1. operational measure of success that is closely related to the achievement of the operational objective being evaluated in the intended operational environment under a specified set of conditions [ISO/IEC/IEEE 24748-4:2016, *Systems and software engineering-Life cycle management-Part 4: Systems engineering planning*, 4.7]

3.2391

measure of performance (MOP)

1. engineering parameter that provides critical performance requirements to satisfy a measure of effectiveness (MOE) [ISO/IEC/IEEE 24748-4:2016, *Systems and software engineering-Life cycle management-Part 4: Systems engineering planning*, 4.8]

Note 1 to entry: An MOP typically characterizes physical or functional attributes relating to the system operation.

3.2392

measurement

1. act or process of assigning a number or category to an entity to describe an attribute of that entity [IEEE 1061-1998 (R2004) *IEEE Standard for Software Quality Metrics Methodology*, 2.7] **2.** assignment of numbers to objects in a systematic way to represent properties of the object **3.** use of a metric to assign a value (e.g., a number or category) from a scale to an attribute of an entity [ISO/IEC 14102:2008 *Information Technology - Guideline for the evaluation and selection of CASE tools*] **4.** set of operations having the object of determining a value of a measure [ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.20] **5.** assignment of values and labels to aspects of software engineering work products, processes, and resources plus the models that are derived from them, whether these models are developed using statistical or other techniques [ISO/IEC TR 19759:2016 *Software Engineering — Guide*

to the Software Engineering Body of Knowledge (SWEBOk), 7] 6. figure, extent, or amount obtained by measuring [IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.7]

3.2393

measurement analyst

1. individual or organization that is responsible for the planning, performance, evaluation, and improvement of measurement [ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.18]

3.2394

measurement experience base

1. data store that contains the evaluation of the information products and the measurement process as well as any lessons learned during the measurement process [ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.19]

3.2395

measurement function

1. algorithm or calculation performed to combine two or more base measures [ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.20] 2. algorithm or calculation performed to combine two or more quality measure elements [ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.7]

3.2396

measurement librarian

1. individual or organization that is responsible for managing the measurement data store(s)

3.2397

measurement method

1. logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale [ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.21] 2. logical organization of operations, described generically, used in measurement [ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.8] 3. logical sequence of operations, described generically, used in the performance of measurements [ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.16]

Note 1 to entry: The type of measurement method depends on the nature of the operations used to quantify an attribute. Two types are distinguished: subjective - quantification involving human judgment; objective - quantification based on numerical rules.

3.2398

measurement model

1. implicit or explicit relationship between a latent variable and its (multi-item) measures [ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.10]

3.2399

measurement procedure

1. set of operations, described specifically, used in the performance of a particular measurement according to a given method [ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.22] 2. logical organization of operations, applied specifically, used in the performance of particular measurements according to a given measurement method [ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.9]

Note 1 to entry: A measurement procedure is usually recorded in a document that is sometimes itself called a "measurement procedure" and is usually in sufficient detail to enable an operator to carry out a measurement without additional information.

3.2400

measurement process

1. process for establishing, planning, performing and evaluating software measurement within an overall project or organizational measurement structure [ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.23] 2. process of establishing, planning, performing and evaluating software measurement within an overall project or organizational measurement structure [ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.18] 3. process for establishing, planning, performing and

evaluating systems and software measurement within an overall project or organizational measurement structure
[*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.21]

3.2401

measurement process owner

1. individual or organization responsible for the measurement process [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.24]

3.2402

measurement sponsor

1. individual or organization that authorizes and supports the establishment of the measurement process
[*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.25]

3.2403

measurement standard

1. standard that describes the characteristics of evaluating a process or product

3.2404

measurement user

1. individual or organization that uses the information products [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.26]

3.2405

measuring instrument

1. device intended to be used to make measurements, alone or in conjunction with supplementary device(s)
[*ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*, 3.6]

3.2406

mechanism

1. in an IDEF0 model, the means used by a function to transform input into output [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.71]

3.2407

mechanism arrow

1. arrow or arrow segment that expresses IDEF0 mechanism [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.72]

Note 1 to entry: That is, an object type set whose instances are used by a function to transform input into output. The arrowhead of a mechanism arrow is attached to the bottom side of a box.

3.2408

mechanism loopback

1. loopback of output from one function to be mechanism for another function in the same diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.73]

3.2409

member product

1. product belonging to the product line [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.15]
cf. application

3.2410

memory

1. addressable storage space in a processing unit and all other internal storage that is used to execute instructions
[*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2411

memory capacity

1. maximum number of items that can be held in a given computer memory; usually measured in words or bytes
cf. channel capacity, storage capacity

3.2412

memory compaction

garbage collection

1. storage allocation technique in which the contents of all allocated storage areas are moved to the beginning of the storage space and the remaining storage blocks are combined into a single block **2.** storage allocation technique in which contiguous blocks of non-allocated storage are combined to form single blocks

3.2413

memory dump

1. display of the contents of all or part of a computer's internal storage, usually in binary, octal, or hexadecimal form

cf. change dump, dynamic dump, postmortem dump, selective dump, snapshot dump, static dump

3.2414

memory map

1. diagram that shows where programs and data are stored in a computer's memory

3.2415

menu

1. list displayed on a screen showing available functions from which a choice can be made [*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 list of options displayed by a data processing system, from which the user can select an action to be initiated [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **3.** (on-screen documentation) list of topics from which the user chooses [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.29]

3.2416

menu by-pass

1. in a menu-driven system, a feature that permits advanced users to perform functions in a command-driven mode without selecting options from the menus

3.2417

menu structure

1. the implementation of a dialog by means of a series of interrelated menus and screens [*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*]

3.2418

menu-driven

1. pertaining to a system or mode of operation in which the user directs the system through menu selections
cf. menu by-pass, command-driven

3.2419

merge

1. to combine different changes to the same file

Note 1 to entry: Many systems follow the optimistic strategy of combining all lines that do not conflict.

3.2420

merge from current

1. to merge changes from the current branch into the stable branch(es)

Note 1 to entry: To avoid disruptive changes in a stable branch, code changes are typically first introduced into the current (development) branch, tested, and then merged back.

3.2421

message

- 1.** communication sent from one object to another [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.110*]
cf. request

Note 1 to entry: Message encompasses requests to meet responsibilities as well as simple informative communications.

3.2422

meta-

- 1.** prefix to a concept to imply definition information about the concept [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Specifically, used to designate the location of an object in the three model layers.

3.2423

meta-attribute

- 1.** definition of a characteristic of a meta-entity or meta-relationship [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Instances of a meta-attribute occur in a model as data values.

3.2424

meta-entity

- 1.** definition of a type of data object that occurs in CDIF models [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Specifically, a meta-entity represents a set of zero or more meta-attributes, stored together to represent a thing, event or concept that has instances in a model.

3.2425

meta-meta-attribute

- 1.** definition of a characteristic of a meta-meta-entity or meta-meta-relationship [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Instances of a meta-meta-attribute occur in a metamodel as meta-data values.

3.2426

meta-meta-entity

- 1.** definition of the behavior and structure of meta-entities, meta-relationships, meta-attributes, or subject areas [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: i.e., a definition of the meta-object definitions used to describe information in models

3.2427

meta-meta-relationship

- 1.** definition of a type of data object that occurs in CDIF metamodels [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: Specifically, a meta-meta-relationship represents the definition of a relationship between instances of meta-meta-entities.

3.2428

meta-object

- 1.** generic term for meta-entities, meta-relationships and meta-attributes [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

3.2429

meta-object facility

MOF

- 1.** specification of the object management group for repositories of type information for arbitrary type systems [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function, 3.3.1*]

3.2430

meta-relationship

- 1.** definition of a type of data object that occurs in CDIF models [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: Specifically, a meta-relationship represents the definition of a relationship between meta-entities that has instances in a model. A meta-relationship can also define a set of zero or more meta-attributes, stored together to represent characteristics of a relationship between meta-entities.

3.2431

metadata

- 1.** data that describe other data [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.29]

3.2432

metalinguage

- 1.** language used to specify some or all aspects of a language
cf. stratified language, unstratified language

EXAMPLE: Backus-Naur form

3.2433

metamodel

- 1.** logical information model that specifies the modeling elements used within another (or the same) modeling notation [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description*, 3.8]
2. metamodel Vm for a subset of IDEFobject is a view of the constructs in the subset that is expressed using those constructs such that there exists a valid instance of Vm that is a description of Vm itself [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.111]
3. model containing detailed definitions of the meta-entities, meta-relationships and meta-attributes whose instances appear in the model section of a CDIF transfer [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]
4. specification of the concepts, relationships and rules that are used to define a methodology [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 3.4]
5. model defining the concepts and their relations for some modeling notation [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.6]
6. special kind of model that specifies the abstract syntax of a modeling language [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.2434

metamodel element

- 1.** element of a meta-model from which model elements are instantiated [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.2435

method

- 1.** implementation of an operation [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.21]
2. statement of how property values are combined to yield a result [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.112]
3. code that is executed to perform a service [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.4.1]

3.2436

method engineer

- 1.** person who designs, builds, extends and maintains methodologies [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 3.10]

Note 1 to entry: Method engineers create methodologies from metamodels via generation.

3.2437

method standard

- 1.** standard that describes the characteristics of the orderly process or procedure used in the engineering of a product or performing a service

3.2438

methodology

- 1.** a system of practices, techniques, procedures, and rules used by those who work in a discipline [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** specification of the process to follow together with the work products to be used and generated, plus the consideration of the people and tools involved, during an IBD development effort [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies, 3.2*]

3.2439

methodology element

- 1.** simple component of a methodology [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies, 3.6*]

Note 1 to entry: Usually, methodology elements include the specification of what tasks, activities, techniques, models, documents, languages and/or notations can or must be used when applying the methodology. Methodology elements are related to each other, comprising a network of abstract concepts. Typical methodology elements are Capture Requirements, Write Code for Methods (kinds of tasks), Requirements Engineering, High-Level Modeling (kinds of activities), Pseudo-code, Dependency Graphs (notations), Class, Attribute (kinds of model building blocks), Class Model, Class Diagram, Requirements Specification (kind of work products).

3.2440

metric

- 1.** quantitative measure of the degree to which a system, component, or process possesses a given attribute **2.** defined measurement method and the measurement scale [*ISO/IEC 14102:2008 Information Technology - Guideline for the evaluation and selection of CASE tools*]
cf. measure, software quality metric

3.2441

metric validation

- 1.** act or process of ensuring that a metric reliably predicts or assesses a quality factor [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.11*]

3.2442

metric value

- 1.** metric output or an element that is from the range of a metric [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.12*]

3.2443

metrics framework

- 1.** decision aid used for organizing, selecting, communicating, and evaluating the required quality attributes for a software system [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.9*] **2.** hierarchical breakdown of quality factors, quality subfactors, and metrics for a software system [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.9*]

3.2444

metrics sample

- 1.** set of metric values that is drawn from the metrics database and used in metrics validation [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.1*]

3.2445

MFLOPS

- 1.** millions of floating point operations per second **2.** megaflops, a unit of measure of processing performance equal to one million floating-point operations per second [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. KOPS, MIPS

Note 1 to entry: a measure of computer processing speed

3.2446

micro code assembler

- 1.** computer program that translates microprograms from symbolic form to binary form

3.2447

microarchitecture

- 1.** microword definition, data flow, timing constraints, and precedence constraints that characterize a given microprogrammed computer

3.2448

microcode

- 1.** collection of microinstructions, comprising part of or all of microprograms [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*]

3.2449

microcomputer

- 1.** digital computer whose processing unit consists of one or more microprocessors, and includes storage and input-output facilities [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. microprogrammable computer

3.2450

microcontroller (unit) (MCU)

single chip microcomputer

- 1.** unit with application control function on a single chip

Note 1 to entry: It contains a processor, RAM, ROM, clock, register, and I/O control unit.

3.2451

microinstruction

- 1.** in microprogramming, an instruction that specifies one or more of the basic operations needed to carry out a machine language instruction
cf. micro code, microoperation, microprogram

Note 1 to entry: Types include diagonal microinstruction, horizontal microinstruction, vertical microinstruction.

3.2452

microoperation

- 1.** in microprogramming, one of the basic operations needed to carry out a machine language instruction
cf. microinstruction

3.2453

microprocessor

- 1.** processor whose elements have been miniaturized into one or a few integrated circuits [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2454

microprogram

- 1.** sequence of instructions, called microinstructions, specifying the basic operations needed to carry out a machine language instruction [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*]

3.2455

microprogrammable computer

- 1.** microprogrammed computer in which microprograms can be created or altered by the user

3.2456

microprogrammed computer

- 1.** computer in which machine language instructions are implemented by microprograms rather than by hard-wired logic

cf. microarchitecture, microcomputer, microprogrammable computer.

3.2457

micropogramming

1. process of designing and implementing the control logic of a computer by identifying the basic operations needed to carry out each machine language instruction and representing these operations as sequences of instructions in a special memory called control store

cf. micro code, microinstruction, micropogram

Note 1 to entry: This method is an alternative to hard-wiring the control signals necessary to carry out each machine language instruction. Techniques include bit steering, compaction, residual control, single-level encoding, two-level encoding.

3.2458

microword

1. addressable element in the control store of a microprogrammed computer

3.2459

middleware

1. software layer between an operating system and the software applications

3.2460

migratability

1. ability to change the configuration, substituting one reference point of an object for another while the object is being used [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 15.4.2]

cf. portability

3.2461

migrated attribute

1. foreign key attribute of a child entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.113]

Note 1 to entry: [key style]

3.2462

migration

1. moving a cluster to a different capsule [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.27]

3.2463

migration transparency

1. distribution transparency which masks from an object, the ability of a system to change the location of that object [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.4.1.4]

Note 1 to entry: Migration is often used to achieve load balancing and reduce latency.

3.2464

milestone

1. a significant point or event in a project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** scheduled event used to measure progress

EXAMPLE: Major milestones for software projects can include an acquirer or managerial sign-off, baselining of a specification, completion of system integration, and product delivery. Minor milestones can include baselining of a software module or completion of a chapter of the user manual.

3.2465

milestone list

1. a list identifying all project milestones and normally indicates whether the milestone is mandatory or optional [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2466

milestone schedule

- 1.** a summary-level schedule that identifies the major schedule milestones [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. master schedule

3.2467

MIM

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

3.2468

minicomputer

- 1.** digital computer that is functionally intermediate between a microcomputer and a mainframe [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: Servers and network devices have generally replaced minicomputers

3.2469

minimax regret rule

- 1.** in decision making under uncertainty, selection of the alternative that minimizes the regret that one would have, if one chose the wrong alternative under each state of nature; that is, the alternative that has the smallest maximum regret

cf. Hurwicz criterion, maximax rule, maximin rule

3.2470

minimum attractive rate of return (MARR)

- 1.** lowest rate of return at which an organization will consider investing; the interest rate used in business decision analysis

cf. opportunity cost

Note 1 to entry: reflects a rate of return that the organization is confident it can achieve through typical activities

3.2471

minimum delay programming

- 1.** programming technique in which storage locations for computer instructions and data are chosen so that access time is minimized

3.2472

minimum tasks

- 1.** those verification and validation tasks required for the integrity level assigned to the system, software, or hardware to be verified and validated [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*] **2.** those tasks required for the integrity level assigned to the software to be tested.

3.2473

MIPS

- 1.** million instructions per second

cf. KOPS, MFLOPS

Note 1 to entry: a measure of computer processing speed

3.2474

mirror site

- 1.** duplicate copy of a master site maintained on a different host typically to provide redundancy, higher performance, or local access [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.18*]

3.2475

MIS

- 1.** management information system

3.2476

mistake

- 1.** human action that produces an incorrect result

Note 1 to entry: The fault tolerance discipline distinguishes between a human action (a mistake), its manifestation (a hardware or software fault), the result of the fault (a failure), and the amount by which the result is incorrect (the error).

3.2477

mixed entry table

- 1.** decision table whose stub consists of rows in which limited and extended entries are written [*ISO 5806:1984 Information processing — Specification of single-hit decision tables*, 3.16]

3.2478

mixed mode

mixed type

- 1.** pertaining to an expression that contains two or more different data types

EXAMPLE: $Y = X + N$, where X and Y are floating point variables and N is an integer variable

3.2479

MMC(S)

- 1.** Multimedia Conferencing (System) [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.2480

MMI

- 1.** man-machine interface

cf. user interface

3.2481

mobility schema

- 1.** specification putting constraints on the mobility of an object [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 16.4.1.1]

3.2482

mock object

- 1.** temporary dummy objects created to aid testing until the real objects become available

3.2483

mock-up

- 1.** throw-away product [*ISO/IEC TR 14759:1999 Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use*, 3.2 d)]

Note 1 to entry: It can be retained for verification or training, and as a record.

3.2484

mode

- 1.** set of related features or functional capabilities of a product **2.** value taken from the transition's type when considering a high-level Petri Net graph [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26.2.1]

EXAMPLE: on-line, off-line, and maintenance modes

3.2485

model

- 1.** representation of a real -world process, device, or concept **2.** representation of something that suppresses certain aspects of the modeled subject [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.115] **3.** interpretation of a theory for which all the axioms of the theory are true [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.115] **4.** related collection of instances of meta-objects, representing (describing or prescribing) an information system, or parts thereof, such as a software product [*ISO/IEC 15474-*

1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2] **5.** semantically closed abstraction of a system or a complete description of a system from a particular perspective **6.** system of postulates, value declarations and inference rules presented as a description of a state of affairs (universe of discourse) *[ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 7.3]* **7.** representation of a system of interest, from the perspective of a related set of concerns *[ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4]*

3.2486

model element

1. instance of a meta-model element *[ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4]*

3.2487

model glossary

1. the collection of the names and definitions of all defined concepts that appear within the views of a model *[IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.116]*

3.2488

model hierarchy

1. diagrams that correspond to the nodes of the hierarchical graph structure of an IDEF0 model *[IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.74]*

3.2489

model kind

1. conventions for a type of modeling *[ISO/IEC/IEEE 42010:2011 Systems and software engineering — Architecture description, 3.9]*

EXAMPLE: balance sheets, class diagrams, data flow diagrams, organization charts, Petri nets, state transition models

3.2490

model layers

1 different layers of definition (or abstraction) used in defining the CDIF family of standards *[ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2]*

Note 1 to entry: The four model layers in CDIF are user data, model, metamodel, meta-metamodel. Any given model layer provides an accurate and complete definition of all the instances that occur one layer below the given layer. For example, the meta-metamodel provides a set of definitions that are used to construct and understand the metamodel; the metamodel provides a set of definitions that are used to construct and understand a model.

3.2491

model name

1. unique, descriptive name that distinguishes one IDEF0 model from other IDEF0 models with which it is associated *[IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.75]*

Note 1 to entry: An IDEF0 model's model name and model name abbreviation are placed in the A-0 context diagram along with the model's purpose statement and viewpoint statement.

3.2492

model name abbreviation

1. unique short form of a model name that is used to construct diagram references *[IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.76]*

3.2493

model note

1. textual and/or graphical component of a diagram that records a fact not otherwise depicted by a diagram's boxes and arrows *[IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.77]*

3.2494

model note number

1. integer number, placed inside a small square, that unambiguously identifies a model note in a specific diagram
[IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.78]

3.2495

model page

1. logical component of an IDEF0 model that can be presented on a single sheet of paper
[IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.79]

Note 1 to entry: Model pages include diagram, text, FEO, and glossary pages.

3.2496

modeling

1. the activity of representing some elements of a process, device, or concept
[Software Extension to the PMBOK® Guide Fifth Edition]

3.2497

modeling tool

1. tool that provides support for modeling, i.e., representing, a software product or an information system
[ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2]

3.2498

modifiability

1. ease with which a system can be changed without introducing defects
2. degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality
[ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.7.4]
cf. analyzability, maintainability, modularity

3.2499

modifiable

1. structured and has a style such that changes can be made completely, consistently, and correctly while retaining the structure
[ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.15]

3.2500

modification request (MR)

1. proposed changes to a product that is being maintained
[IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.6] **2.** forms associated with the various trouble/problem-reporting documents and the configuration change control documents
cf. change request, request for change

Note 1 to entry: The MR can be classified as a correction or enhancement and identified as corrective, preventive, adaptive, or perfective maintenance. Trouble/problem-reporting documents include incident and problem reports; configuration change control documents include software change requests (SCRs).

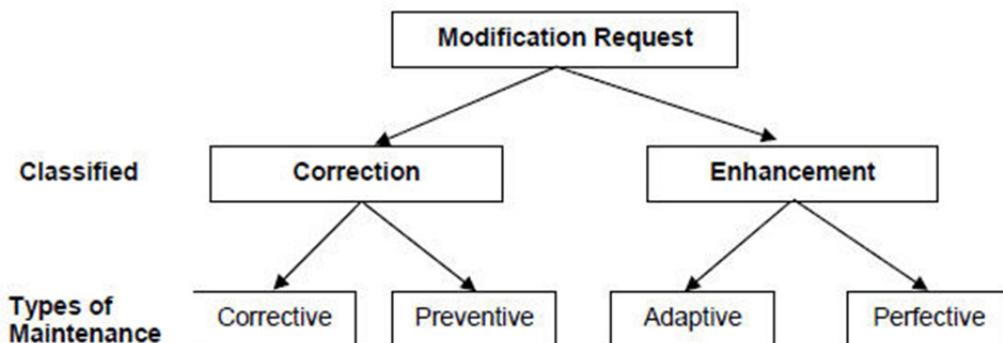


Figure 16 — Modification request

3.2501

modified source statement

- 1.** source statement that has been changed from the original source

3.2502

modified-off-the-shelf (MOTS)

- 1.** software product that is already developed and available, usable either 'as is' or with modification, and provided by the supplier, acquirer, or a third party [*IEEE 1062-2015 IEEE Recommended Practice for Software Acquisition, 3.1*]

3.2503

MODL

- 1.** Meta-Object Definition Language [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function, 4*]

3.2504

modular

- 1.** composed of discrete parts

cf. modular decomposition, modular programming

3.2505

modular decomposition

modularization

- 1.** process of breaking a system into components to facilitate design and development; an element of modular programming

cf. cohesion, coupling, demodularization, factoring, functional decomposition, hierarchical decomposition, packaging

3.2506

modular programming

- 1.** software development technique in which software is developed as a collection of modules
cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transaction analysis, transform analysis

3.2507

modularity

- 1.** degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.7.1*] **2.** software attributes that provide a structure of highly independent components
cf. cohesion, coupling, modifiability

3.2508

module

- 1.** program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*] **2.** logically separable part of a program [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*] **3.** set of source code files under version control that can be manipulated together as one **4.** collection of both data and the routines that act on it

Note 1 to entry: The terms 'module', 'component,' and 'unit' are often used interchangeably or defined to be subelements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized.

3.2509

module data

instance data

class data

1. data that can be accessed by any routine within the module in which it is declared but not by routines in other modules

3.2510

MOE

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

3.2511

MOF

1. meta-object facility [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function, 4; ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications, 4*]

3.2512

monadic selective construct

1. if-then-else construct in which processing is specified for only one outcome of the branch, the other outcome resulting in skipping this processing

cf. dyadic selective construct

3.2513

monitor

execution monitor

1. software tool or hardware device that operates concurrently with a system or component and supervises, records, analyzes, or verifies the operation of the system or component **2.** collect project performance data with respect to a plan, produce performance measures, and report and disseminate performance information [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. hardware monitor, software monitor

3.2514

monitor and control project work

1. the process of tracking, reviewing, and regulating the progress to meet the performance objectives defined in the project management plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2515

monitoring

1. examination of the status of the activities of a supplier and of their results by the acquirer or a third party [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.19*]

3.2516

monitoring and controlling process group

1. those processes required to track, review, and regulate the progress and performance of the project; identify any areas in which changes to the plan are required; and initiate the corresponding changes [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2517

monolithic executor

1. executor consisting of a single artifact [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.2518

Monte Carlo analysis

1. a technique that computes, or iterates, the project cost or project schedule many times using input values selected at random from probability distributions of possible costs or durations, to calculate a distribution of possible total project cost or completion dates [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2519

Monte Carlo simulation

- 1.** a process which generates hundreds or thousands of probable performance outcomes based on probability distributions for cost and schedule on individual tasks. The outcomes are then used to generate a probability distribution for the project as a whole [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2520

MOP

- 1.** measure of performance [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.2*]

3.2521

MOPS

- 1.** million operations per second

3.2522

most likely duration

- 1.** estimate of the most probable activity duration that takes into account all of the known variables that could affect performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2523

MOTS

- 1.** modified-off-the-shelf [*IEEE 1062-2015 IEEE Recommended Practice for Software Acquisition, 3.1*]

3.2524

MOU

- 1.** memorandum of understanding

3.2525

move

- 1.** to read data from a source, altering the contents of the source location, and to write the same data elsewhere in a physical form that can differ from that of the source

cf. copy

EXAMPLE: to move data from one file to another

3.2526

MPa

- 1.** measure IT project activity [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking, 4*]

3.2527

MPLa

- 1.** manage benchmarking program level activity [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking, 4*]

3.2528

MRa

- 1.** maintain repository activity [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking, 4*]

3.2529

MTBF

- 1.** mean time between failures

3.2530

MTP

- 1.** master test plan.

3.2531

MTR

1. master test report

3.2532

MTTR

1. mean time to repair

3.2533

multi-attribute decision

multiple-attribute decision

1. decision that considers more than just one criterion

EXAMPLE: a decision based on price, delivery date, and quality all at the same time

3.2534

multi-component system

1. measured system consisting of more than one piece of software [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, A.13; *ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*, A.10]

3.2535

multi-core

1. chip with two or more microprocessor units

3.2536

multi-core processor

1. single integrated circuit chip with more than one processing unit

3.2537

multi-criteria decision analysis

1. This technique utilizes a decision matrix to provide a systematic analytical approach for establishing criteria, such as risk levels, uncertainty, and valuation, to evaluate and rank many ideas [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2538

multi-level cache

1. layered cache with progressively larger size and slower access

Note 1 to entry: denoted by L1 (level 1) to L4 (level 4) cache, where the first level is the fastest and smallest memory size for immediate access, and the highest level is slowest, but the largest memory size

3.2539

multi-valued

1. a mapping that is not a function [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.117]
cf. function

3.2540

multi-valued property

1. a property with a multi-valued mapping [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.118]
cf. single-valued property

3.2541

multiaddress instruction

multiple-address instruction

1. computer instruction that contains more than one address field

cf. one-address instruction

3.2542

multidimensional construct

1. construct that consists of a number of unidimensional constructs [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.11*]

3.2543

multiple inclusive selective construct

1. special instance of the case construct in which two or more different values of the control expression result in the same processing

EXAMPLE: values 1 and 2 cause one branch, 3 and 4 cause another, and so on

3.2544

multiple inheritance

1. ability of a subclass to inherit responsibilities from more than one superclass [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.119*] 2. situation when the subtype inherits all of the meta-attributes and meta-relationships of all of its supertypes (and their supertypes) [*ISO/IEC 15474-2:2002 Information technology — CDIF framework — Part 2: Modelling and extensibility, 6.2.6*]

3.2545

multiple readers and writers

1. algorithm that lets multiple readers access a shared data repository concurrently; however, writers must have mutually exclusive access to update the repository

3.2546

multiple-criteria decision making (MCDM)

multi-attribute decision making

1. making preference decisions (e.g., evaluation, prioritization, and selection) of available alternatives characterized by multiple criteria [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.9*]

Note 1 to entry: An MCDM with one alternative is the same as the development of a composite measure.

3.2547

multiple-hit decision table

1. decision table where at least one set of conditions will be satisfied by more than one rule [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.3*]

3.2548

multiplex receptacle

1. specialization of a receptacle that allows multiple simultaneous connections [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.2549

multiplicity

1. natural number (i.e., non-negative integer) which describes the number of repetitions of an item in a multiset [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.15.1*]

3.2550

multiprocessing

1. mode of operation in which two or more processes are executed concurrently by separate processing units that have access (usually) to a common main storage
cf. multiprogramming, multitasking, time sharing

3.2551

multiprogramming

1. mode of operation in which two or more computer programs are executed in an interleaved manner by a single processing unit

cf. multiprocessing, multitasking, time sharing

3.2552

multiset

- 1.** collection of items where repetition of items is allowed [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.15]

3.2553

multiset cardinality

cardinality of a multiset

- 1.** sum of the multiplicities of each of the members of the multiset [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.15.2]

3.2554

multitasking

- 1.** mode of operation in which two or more tasks are executed in an interleaved manner
cf. multiprocessing, multiprogramming, time sharing

3.2555

mutable class

- 1.** class for which the set of instances is not fixed; its instances come and go over time. [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.120]
cf. immutable class, state class

3.2556

mutation testing

- 1.** testing methodology in which two or more program mutations are executed using the same test cases to evaluate the ability of the test cases to detect differences in the mutations

3.2557

mutual exclusion

- 1.** giving access to shared data only to one task at a time

Note 1 to entry: can be enforced by means of binary semaphores or by using monitors.

3.2558

mutually exclusive clustering

- 1.** task structuring criterion in which a group of objects are combined into one task because only one object can be executed at any one time

3.2559

N 2 diagram

- 1.** system engineering or software engineering tool for tabulating, defining, analyzing, and describing functional interfaces and interactions among system components

Note 1 to entry: The N 2 diagram is a matrix structure that graphically displays the bidirectional interrelationships between functions and components in a given system or structure.

3.2560

n-address instruction

- 1.** computer instruction that contains n address fields, where n is any non-negative integer
cf. one-address instruction, two-address instruction, n-plus-one address instruction

3.2561

N-ary relationship

- 1.** relationship with arity (degree) $n > 2$ [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: A relationship that has more than two participating entities. (Note that a single entity can participate several times in a single relationship.)

3.2562

n-level address

1. indirect address that specifies the first of a chain of n storage locations, the first n-1 of which contains the address of the next location in the chain and the last of which contains the desired operand
cf. direct address, immediate data

EXAMPLE: a two-level address

3.2563

n-plus-one address instruction

1. computer instruction that contains n+1 address fields, the last containing the address of the instruction to be executed next
cf. one-plus-one address instruction, two-plus-one address instruction, n-address instruction

3.2564

N/A

1. not applicable [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.2*] **2.** not available

3.2565

name

1. word or phrase that designates some model construct [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.121*] **2.** term which, in a given naming context, refers to an entity [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 12.1*]

Note 1 to entry: Such as a class, responsibility, subject domain, etc.

3.2566

name resolution

1. process by which, given an initial name and an initial naming context, an association between a name and the entity designated by the initial name can be found [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 12.8*]

3.2567

name space

1. set of terms usable as names [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 12.3*]

3.2568

named constant

1. identifier that refers to a numeric or string value that does not change during program execution

3.2569

named constraint

1. constraint that is specific to a particular model, rather than being inherent in some modeling construct [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.122*]

Note 1 to entry: Such as a cardinality constraint. A named constraint is explicitly named, its meaning is stated in natural language, and its realization is written in the specification language.

3.2570

naming action

1. action that associates a term from a name space with a given entity [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 12.5*]

3.2571

naming context

1. relation between a set of names and a set of entities [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 12.4*]

3.2572

naming domain

1. subset of a naming context such that all naming actions are performed by the controlling object of the domain (the name authority object) [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 12.6]

3.2573

naming graph

1. directed graph where each vertex denotes a naming context, and where each edge denotes an association between a name appearing in the source naming context and the target naming context [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 12.7]

3.2574

nano code

1. collection of nanoinstructions

3.2575

nanoinstruction

1. in a two-level implementation of microprogramming, an instruction that specifies one or more of the basic operations needed to carry out a microinstruction

3.2576

nanostore

1. in a two-level implementation of microprogramming, a secondary control store in which nanoinstructions reside

3.2577

natural language

1. language whose rules are based on usage rather than being pre-established prior to the language's use 2. language whose rules are based on current usage without being specifically prescribed [ISO/IEC 2382:2015, *Information technology — Vocabulary*] cf. formal language

EXAMPLE: German and English

3.2578

navigation

1. means by which a user moves from one part of a software application to another 2. process of accessing on-screen documentation and moving between different items of information [ISO/IEC 26513:2009 *Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.23] 3. process of accessing on-screen information by moving between different locations in a website or electronic document [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.19]

3.2579

navigational aids

1. features of software that help the user to navigate around a computer application [ISO/IEC 20968:2002 *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10]

EXAMPLE: shortcut keys to move through a dialogue faster

3.2580

NDI

1. non-developmental item.

3.2581

near-critical activity

1. a schedule activity that has low total float. The concept of near-critical is equally applicable to a schedule activity or schedule network path. The limit below which total float is considered near critical is subject to expert judgment and varies from project to project [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition]

3.2582

negotiated settlements

- 1.** the process of reaching final equitable settlement of all outstanding issues, claims, and disputes through negotiation [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2583

negotiation

- 1.** the process and activities to resolving disputes through consultations between involved parties [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2584

nest

- 1.** to incorporate a computer program construct into another construct of the same kind

EXAMPLE: to nest one subroutine, block, or loop within another; to nest one data structure within another

3.2585

nesting

- 1.** embedding one construct inside another

EXAMPLE: to embed a WHILE loop within another WHILE loop or to embed an IF test within a WHILE loop

3.2586

net

- 1.** general term used to describe all classes of Petri Nets [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.16*]

3.2587

net graph

- 1.** directed graph comprising a set of nodes of two different kinds, called places and transitions, and their interconnection by directed edges, called arcs, such that only places can be connected to transitions, and transitions to places, but never transitions to transitions, nor places to places [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.16.1*]

3.2588

network

- 1.** arrangement of nodes and interconnecting branches [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2589

network chart

- 1.** directed graph used for describing and scheduling events, activities, and their relationships in project control [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2590

network logic

- 1.** the collection of schedule activity dependencies that makes up a project schedule network diagram [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2591

network open end

- 1.** schedule activity without any predecessor activities or successor activities, causing a break in a schedule network path

Note 1 to entry: Network open ends are usually caused by missing logical relationships

3.2592

network path

- 1.** any continuous series of schedule activities connected with logical relationships in a project schedule network diagram [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2593

network planning

1. technique that uses network charts for planning, scheduling, and controlling a project [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2594

networking

1. developing relationships with persons who can assist in the achievement of objectives and responsibilities **2.** establishing connections and relationships with other people from same or other organizations [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2595

new source statements

1. sum of the added and modified source statements

3.2596

no-op

1. no-operation

3.2597

no-operation

do-nothing operation

1. computer operation whose execution has no effect except to advance the instruction counter to the next instruction

Note 1 to entry: used to reserve space in a program or, if executed repeatedly, to wait for a given event; often abbreviated no-op

3.2598

node

1. in a diagram, a point, circle, or other geometric figure used to represent a state, event, or other item of interest **2.** configuration of engineering objects forming a single unit for the purpose of location in space, and which embodies a set of processing, storage and communication functions [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.7*] **3.** modeled function located within the hierarchical graph structure of an IDEF0 model by its designated node number [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.80*] **4.** one of the defining points of a schedule network; a junction point joined to some or all of the other dependency lines [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **5.** vertex of a net graph, i.e., a place or a transition [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.16.2*] *cf.* graph (2)

EXAMPLE: a computer and the software it supports (operating system and applications)

Note 1 to entry: A node can have internal structure which is not of concern in an engineering specification.

3.2599

node index

1. text listing, often indented, of the nodes in an IDEF0 model, shown in outline order [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.81*]

Note 1 to entry: Same meaning and node content as a node tree.

3.2600

node letter

1. letter that is the first character of a node number [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.82*]

3.2601

node number

1. expression that unambiguously identifies a function's position in a model hierarchy [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.83*]

Note 1 to entry: A node number is constructed by concatenating a node letter, the diagram number of the diagram that contains the box that represents the function, and the box number of that box.

3.2602

node tree

1. graphical listing of the nodes of an IDEF0 model, showing parent-child relationships as a graphical tree [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.84*]

Note 1 to entry: Same meaning and node content as a node index.

3.2603

nomenclature standard

1. standard that describes the characteristics of a system or set of names, or designations, or symbols

3.2604

nominal group technique

1. a technique that enhances brainstorming with a voting process used to rank the most useful ideas for further brainstorming or for prioritization [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2605

nominal scale

1. scale in which the measurement values are categorical
cf. ordinal scale, interval scale, ratio scale

EXAMPLE: For example, the classification of defects by their type does not imply order among the categories.

3.2606

non-compensatory model

1. multiple-criteria decision-making model that does not allow criteria to compensate for each other in proportion to their weights [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.12*]

Note 1 to entry: Strongly positive or negative terms influence the overall composite value disproportionately, although the weight stays the same. There are various non-compensatory models depending on the evaluation policy, the purpose of the composite measure, or the measurement scale.

3.2607

non-deliverable item

1. hardware or software product that is not required to be delivered under the contract but may be employed in the development of a product [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.20*]

3.2608

non-primary entity

1. a data entity-type arrived at by Third Normal Form analysis which is not one of the main entity-types for which the application in question has been built [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

cf. system entity

Note 1 to entry: Non-primary entities have only very few attributes, e.g. code, description

3.2609

non-repudiation

1. degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.6.3]

3.2610

non-terminal symbol

1. part of the hierarchical definition of a syntax that is further decomposed in the hierarchy [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

3.2611

non-time-critical computationally intensive task

1. low-priority compute-bound task that consumes spare CPU cycles

3.2612

non-volatile memory

1. unit that stores data whether power is on or off

3.2613

noncompensatory decision technique

1. a multi-attribute decision technique that weighs all attributes equally, without allowing lower performance in one attribute to be traded off against better performance in another attribute
cf. compensatory decision technique

3.2614

nonconformance work

1. In the cost of quality framework, non-conformance work is done to deal with the consequences of errors and failures in doing activities correctly on the first attempt. In efficient quality management systems, the amount of non-conformance work will approach zero [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition]

3.2615

nonconformity

1. non-fulfillment of a requirement

3.2616

nondelivered source statement

1. source statement that is developed in support of the final product, but not delivered to the customer

3.2617

nondestructive read

1. read operation that does not erase the data in the accessed location

cf. destructive read

3.2618

nondeveloped source statement

1. existing source statement that is reused or deleted

3.2619

nondevelopmental

1. developed prior to its current use in an acquisition or development process

Note 1 to entry: Such an item can require minor modifications to meet the requirements of its current intended use.

3.2620

nondimensional scaling

1. decision technique in which attribute values are converted into a common scale where they can be added together to make a composite score for each alternative

cf. additive weighting, analytic hierarchy process, compensatory decision technique

3.2621

nonfunctional requirement

performance attribute

1 software requirement that describes not what the software will do but how the software will do it
cf. design constraint, functional requirement, performance requirement, quality requirement

EXAMPLE: software performance requirements, software external interface requirements, software design constraints, and software quality attributes.

3.2622

nonidentifying relationship

1. a specific (not many-to-many) relationship in which some or all of the attributes contained in the primary key of the parent entity do not participate in the primary key of the child entity [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.123]
cf. identifying relationship, mandatory nonidentifying relationship, optional nonidentifying relationship [key style]

3.2623

nonintrinsic relationship

1. relationship that is partial, multi-valued, or changeable [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.125]
cf. intrinsic relationship

3.2624

nonkey attribute

1. attribute that is not the primary or a part of a composite primary key of an entity [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.125]

Note 1 to entry: [key style]

3.2625

nonprocedural language

1. language in which the user states what is to be achieved without having to state specific instructions that the computer must execute in a given sequence

cf. procedural language, declarative language, interactive language, rule-based language

3.2626

nonprocedural programming language

1. computer programming language used to express the parameters of a problem rather than the steps in a solution.

cf. procedural programming language

EXAMPLE: report writer or sort specification languages

3.2627

nontechnical requirement

1. requirement affecting product and service acquisition or development that is not a property of the product or service

EXAMPLE: numbers of products or services to be delivered, data rights for delivered COTS nondevelopmental items, delivery dates, milestones with exit criteria, work constraints associated with training, site provisions, and deployment schedules.

3.2628

NOR

1. in configuration management, a notice of revision

3.2629

normalization

1. process by which a data structure can be transformed by a database designer into a set of relations that have no repeating groups

3.2630

not printable

- 1.** not a <GeneralPrintableChar>, ", #,], <EscapeCharacter> or <WhiteSpace> [ISO/IEC 15475-3:2002 *Information technology — CDIF transfer format — Part 3: Encoding ENCODING.1*, 7.2.11]

3.2631

notation

- 1.** means of concrete representation for a particular type of a model, expressed as a grammar and suitable glyphs for its terminal symbols [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 7.5]

3.2632

notation standard

- 1.** standard that describes the characteristics of formal interfaces within a profession

3.2633

note

- 1.** helpful hint or other information that assists the user by emphasizing or supplementing important points of the main text [ISO/IEC 26513:2009 *Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.24] **2.** body of free text that describes some general comment or specific constraint about a portion of a model [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.126]

cf. warning, caution

3.2634

notebook computer

laptop computer

- 1.** battery-powered portable computer small and light enough to be operated anywhere [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.2635

notice of revision (NOR)

- 1.** form used in configuration management to propose revisions to a drawing or list, and, after approval, to notify users that the drawing or list has been, or will be, revised accordingly

cf. configuration control, engineering change, specification change notice

3.2636

nucleus

- 1.** engineering object which coordinates processing, storage and communications functions for use by other engineering objects within the node to which it belongs [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.6]

cf. kernel

EXAMPLE: an operating system

3.2637

numeric

- 1.** pertaining to data that consists of numerals as well as functional units that use the data [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.2638

O&S

- 1.** operations and support [IEEE 15288.2:2014 *IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2639

object

- 1.** encapsulation of data and services that manipulate that data [ISO/IEC 19500-1:2012 *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.3.1] **2.** structure of encapsulated behaviors with visible interactions and messages [IEEE 1175.3-2004 *IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*] **3.** specific entity that exists in a

program at runtime, in object-oriented programming **4.** member of an object set and an instance of an object type [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 2.1.85] **5.** model of an entity [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 8.1] **6.** arc, node, reference node, or page of a net graph [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.7]
cf. object code, object module, object program

Note 1 to entry: An object represents something in the observable world that is distinguished from other instances of its object type and can be uniquely identified.

3.2640

object code

- 1.** computer instructions and data definitions in a form output by an assembler or compiler
cf. source code

Note 1 to entry: An object program is made up of object code.

3.2641

object identifier

OID

- 1.** some concrete representation for the identity of an object (instance) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.128]

Note 1 to entry: The object identifier (OID) is used to show examples of instances with identity, to formalize the notion of identity, and to support the notion in programming languages or database systems.

3.2642

object implementation

- 1.** definition that provides the information needed to create an object and to allow the object to participate in providing an appropriate set of services [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.4.2]

3.2643

Object Management Group (OMG)

- 1.** international standards organization that owns and maintains CORBA and UML standards

3.2644

object module

- 1.** computer program or subprogram that is the output of an assembler or compiler
cf. load module, object program

3.2645

object of interest (-type)

- 1.** any thing that is identified from the point of view of the functional user requirements about which the software is required to process or store data. [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method*, 2.19]

Note 1 to entry: An object of interest can be any physical thing, as well as any conceptual object or part of a conceptual object in the world of the functional user.

3.2646

object program

target program

- 1.** computer program that is the output of an assembler or compiler
cf. source program object module

3.2647

object reference

- 1.** value that unambiguously identifies an object. [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.24]

Note 1 to entry: Object references are never reused to identify another object.

3.2648

object set

1. subset of instantiations from the set of all possible instantiations of all object types within an object type set. [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.86*]

Note 1 to entry: An object set is a subset of the union of the members of an object type set; the set of object sets includes the empty set and the set of the union of the members of the object type set itself. An object set is modeled by an arrow segment.

3.2649

object type

1. set of all possible instantiations of a singular concept, either physical or data, within an IDEF0 model. [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.87*] **2.** type whose members are object references. [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces, 5.3.4*]

Note 1 to entry: An IDEF0 object type is generally analogous to an IDEF1X entity or an IDEF1 entity class.

3.2650

object type set

1. named set of one or more object types. [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.88*]

Note 1 to entry: An object type set can include object types that are themselves grouped as object type sets. An object type set is designated by an arrow label.

3.2651

object-oriented design

1. software development technique in which a system or component is expressed in terms of objects and connections between those objects

cf. data structure-centered design, input-process-output, modular decomposition, rapid prototyping, stepwise refinement, structured design, transaction analysis, transform analysis

3.2652

object-oriented language

1. programming language that allows the user to express a program in terms of objects and messages between those objects

EXAMPLE: Smalltalk, LOGO

3.2653

objective

purpose

1. something toward which work is to be directed, a strategic position to be attained, or a purpose to be achieved, a result to be obtained, a product to be produced, or a service to be performed. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** practical advantage or intended effect, expressed as preferences about future states. [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.2.1*]

Note 1 to entry: Some objectives are ongoing; some are achieved once met. Objectives can have different aspects (such as financial, health and safety, and environmental goals) and can apply at different levels (such as strategic, organization-wide, project, product and process). The enterprise language systematically uses the term objective, (rather than purpose) and emphasizes the need of expressing objective in measurable terms.

3.2654

objective evidence

1. data supporting the existence or verity of something. [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.2.13*]

Note 1 to entry: Objective evidence can be obtained through observation, measurement, test, or other means. [ISO 9000:2015]

3.2655

objective function

- 1.** formula that relates a decision variable to either the cost or the revenue of an alternative
cf. cost function, income function

3.2656

objref

- 1.** object reference. [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.25]

3.2657

obligation

- 1.** prescription that a particular behavior is required. [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 11.2.4] **2.** collaboration pattern of interaction between two units used when one unit is not independently capable of providing its service behavior without the assistance of the other unit. [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.18]

3.2658

OBS

- 1.** organizational breakdown structure. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2659

observation

- 1.** instance of applying a measurement procedure to produce a value for a base measure. [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.28]

3.2660

observation period

- 1.** time interval, where the measurement procedure is observed for collecting (logging) measurement results for rating or validation, consisting of the rating interval and the supplementary run. [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems*, 4.11]

3.2661

observations

- 1.** a technique that provides a direct way of viewing individuals in their environment performing their jobs or tasks and carrying out processes. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2662

occupational title standard

- 1.** standard that describes the characteristics of the general areas of work or profession

3.2663

OCD

- 1.** operational concept document. [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]
cf. concept of operations (ConOps) document

3.2664

OCL

- 1.** Object Constraint Language. [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function*, 4; *ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications*, 4]

3.2665

octet

8-bit byte

- 1.** byte that consists of eight bits [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2666

ODP

- 1.** Open Distributed Processing [*ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications, 4*]

3.2667

ODP function

- 1.** function required to support Open Distributed Processing [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 4.3.1*]

3.2668

ODP IDL

- 1.** Open Distributed Processing Interface Definition Language [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 4*]

3.2669

ODP standard

- 1.** standard that complies with the ODP Reference Model, directly or indirectly. [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 3.2.3*]

3.2670

ODP system

- 1.** system which conforms to the requirements of ODP standards. [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 3.2.5*]

3.2671

ODP-RM

RM-ODP

- 1.** Open Distributed Processing: Reference Model. [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function, 4*]

3.2672

OEM

- 1.** original equipment manufacturer. [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.2*]

3.2673

off-the-shelf

- 1.** already developed and available. [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.21*]

3.2674

office automation (OA)

- 1.** integration of office activities by means of an information processing system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: This term includes in particular the processing and communication of text, images, and voice.

3.2675

offline

- 1.** pertaining to a device or process that is not under the direct control of the central processing unit of a computer

- 2.** pertaining to the operation of a functional unit that takes place either independently of, or in parallel with, the main operation of a computer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. online

3.2676

offset

relocation factor

- 1.** difference between the loaded origin and the assembled origin of a computer program **2.** number that must be added to a relative address to determine the address of the storage location to be accessed

3.2677

OGC

1. Office of Government Commerce (UK) [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 3]

3.2678

OID

1. object identifier [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service*, 4]

3.2679

OMG

1. Object Management Group [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*; *ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications*, 4]

3.2680

OMT

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

3.2681

on-chip oscillator

1. electronic circuit on a microcomputer that produces a periodic electronic signal, often used for a device clock

3.2682

on-demand scheduling

1. a scheduling approach in which work is pulled from a backlog according to the perceived value to customers and is assigned as resources become available [*Software Extension to the PMBOK® Guide Fifth Edition*] cf. late binding

3.2683

on-screen documentation

1. documentation that is intended to be read on the computer screen by the user while using the software [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.31]
cf. printed documentation, embedded documentation

EXAMPLE: pop-up help and help text on a screen

3.2684

one-address instruction

single-address instruction

single-operand instruction

1. computer instruction that contains one address field

cf. multiaddress instruction, two-address instruction, three-address instruction, four-address instruction, zero-address instruction

EXAMPLE: an instruction to load the contents of location A

3.2685

one-ahead addressing

1. method of implied addressing in which the operands for a computer instruction are understood to be in the storage locations following the locations of the operands used for the last instruction executed
cf. repetitive addressing

3.2686

one-plus-one address instruction

1. computer instruction that contains two address fields, the second containing the address of the instruction to be executed next

cf. two-plus-one address instruction, three-plus-one address instruction, four-plus-one address instruction

EXAMPLE: an instruction to load the contents of location A, then execute the instruction at location B

3.2687

one-time programming (OTP)

- 1.** method of recording data in ROM which can only be written once

3.2688

one-to-many relationship

- 1.** relationship between two state classes in which each instance of one class, referred to as the child class, is specifically constrained to relate to no more than one instance of a second class, referred to as the parent class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.131*]

3.2689

online

- 1.** pertaining to a system or mode of operation in which input data enter the computer directly from the point of origin or output data are transmitted directly to the point where they are used **2.** pertaining to a device or process that is under the direct control of the central processing unit of a computer **3.** pertaining to the operation of a functional unit when under the control of a computer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. batch, conversational, real time

EXAMPLE: an airline reservation system

3.2690

online documentation

- 1.** information accessed by the user through the use of software

Note 1 to entry: can be context-sensitive

3.2691

ontology

- 1.** logical structure of the terms used to describe a domain of knowledge, including both the definitions of the applicable terms and their relationships [*IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.9*]

3.2692

OOD

- 1.** object-oriented design

3.2693

OPA

- 1.** organizational process asset

3.2694

open distributed processing

- 1.** distributed processing designed to conform to ODP standards [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 3.2.4*]

3.2695

open subroutine

direct insert subroutine

- 1.** subroutine that is copied into a computer program at each place that it is called

cf. closed subroutine, inline code, macro

3.2696

operability

- 1.** degree to which a product or system has attributes that make it easy to operate and control [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.4.3*]

Note 1 to entry: Operability corresponds to controllability, (operator) error tolerance, and conformity with user expectations as defined in ISO 9241-110.

3.2697

operable

1. state of

3.2698

operand

1. variable, constant, or function upon which an operation is to be performed

EXAMPLE: In the expression $A = B + 3$, B and 3 are the operands.

3.2699

operating environment (software)

1. set of software operating concurrently on a specified computer system [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.20*]

3.2700

operating mode

1. type of operation for a system

EXAMPLE: high-speed mode, low power mode

3.2701

operating system

1. collection of software, firmware, and hardware elements that controls the execution of computer programs and provides such services as computer resource allocation, job control, input/output control, and file management in a computer system

3.2702

operation

1. in computer mathematics, the action specified by an operator on one or more operands **2.** in programming, a defined action that can be performed by a computer system **3.** running a computer system in its intended environment to perform its intended functions **4.** interaction between a client object and a server object which is either an interrogation or an announcement [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.2*] **5.** property that is a mapping from the (cross product of the) instances of the class and the input argument types to the (cross product of the) instances of the other (output) argument types [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.132*] **6.** action needed to perform an activity [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.11*] **7.** arithmetic or logical operation performed in an algorithmic and manipulation BFC [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, 3.7*] **8.** identifiable entity that denotes the indivisible primitive of service provision that can be requested [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces, 5.3.8*]

EXAMPLE: In the expression $A = B + 3$, the process of adding B to 3 to obtain A.

Note 1 to entry: An operation can consist of other operations.

3.2703

operation and maintenance costs

1. costs associated with using an asset as well as costs of keeping it in a usable condition

3.2704

operation and maintenance phase

1. period of time in the software life cycle during which a software product is employed in its operational environment, monitored for satisfactory performance, and modified as necessary to correct problems or to respond to changing requirements

3.2705

operation code

op code

1. character or set of characters that specifies a computer operation

EXAMPLE: the code BNZ to designate the operation 'branch if not zero'.

3.2706

operation exception

1. exception that occurs when a program encounters an invalid operation code

cf. addressing exception, data exception, overflow exception, protection exception, underflow exception

3.2707

operation field

function field

operation part

1. field of a computer instruction that specifies the operation to be performed

cf. address field

3.2708

operation interface

1. interface in which all the interactions are operations [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.7*]

3.2709

operation interface signature

1. interface signature for an operation interface [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.12*]

Note 1 to entry: An operation interface signature comprises a set of announcements and interrogation signatures as appropriate, one for each operation type in the interface, together with an indication of causality (client or server, but not both) for the interface as a whole, with respect to the object which instantiates the template.

3.2710

operational

1. pertaining to a system or component that is ready for use in its intended environment **2.** pertaining to a system or component that is installed in its intended environment **3.** pertaining to the environment in which a system or component is intended to be used.

3.2711

operational concept (OpsCon)

1. verbal and graphic statement of an organization's assumptions or intent in regard to an operation or series of operations of a system or a related set of systems [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.26; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.25; ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.14*]
cf. concept of operations

Note 1 to entry: The operational concept is designed to give an overall picture of the operations using one or more specific systems, or set of related systems, in the organization's operational environment from the users' and operators' perspective.

3.2712

operational product

1. product which functions in real conditions of operations [*ISO/IEC TR 14759:1999 Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use, 3.2 e*)]

3.2713

operational scenario

1. description of an imagined sequence of events that includes the interaction of the product or service with its environment and users, as well as interaction among its product or service components [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.15*]

Note 1 to entry: Operational scenarios are used to evaluate the requirements and design of the system and to verify and validate the system.

3.2714

operational testing

- 1.** testing conducted to evaluate a system or component in its operational environment.
cf. development testing, acceptance testing, qualification testing

3.2715

operations

- 1.** ongoing execution of activities that produce the same product or provide a repetitive service

EXAMPLE: production, manufacturing, accounting

3.2716

operator

- 1.** individual or organization that performs the operations of a system [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.27; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.26] **2.** entity that performs the operations of a system [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.29] **3.** mathematical or logical symbol that represents an action to be performed in an operation **4.** symbol representing the name of a function [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.17] **5.** individual or an organization that contributes to the functionality of a system and draws on knowledge, skills, and procedures to contribute the function **6.** individual or organization that operates the system [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.4]
cf. secondary user

Note 1 to entry: The role of operator and the role of user can be vested, simultaneously or sequentially, in the same individual or organization. An individual operator combined with knowledge, skills and procedures can be considered as an element of the system. An operator can perform operations on a system that is operated, or of a system that is operated, depending on whether or not operating instructions are placed within the system boundary.

3.2717

operator manual

operations manual

- 1.** document that provides the information necessary to initiate and operate a system or component
cf. diagnostic manual, installation manual, programmer manual, support manual, user manual

Note 1 to entry: Typically described are procedures for preparation, operation, monitoring, and recovery. An operator manual is distinguished from a user manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose.

3.2718

opportunity

- 1.** a risk that would have a positive effect on one or more project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. threat

3.2719

opportunity cost

- 1.** implicit cost associated with investing money in a certain activity, so that it is no longer available for investing elsewhere
cf. minimum attractive rate of return (MARR)

Note 1 to entry: Making an investment means that the same money cannot be invested elsewhere, where it could be earning the MARR

3.2720

opportunity study

- 1.** study to examine a problem and determine whether or not it requires being solved during the time period under consideration. [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2721

OpsCon

- 1.** operational concept [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

3.2722

optical disc (OD)

- 1.** disk which stores binary data in the form of pits which interrupt the reflection of light from a laser

3.2723

optimistic duration

- 1.** estimate of the shortest activity duration that takes into account all of the known variables that could affect performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2724

optimization analysis

- 1.** balance of competing components to achieve the best performance under the situation

Note 1 to entry: For example, an algorithm that runs faster will typically use more memory. Optimization balances the value of a faster run time against the cost of additional memory.

3.2725

optimizing process

- 1.** quantitatively managed process that is improved based on an understanding of the common causes of variation inherent in the process.

Note 1 to entry: The focus of an optimizing process is on continually improving the range of process performance through both incremental and innovative improvements.

3.2726

optional

- 1.** syntax keyword used to specify a partial mapping [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.133]
cf. mandatory, partial

3.2727

optional attribute

- 1.** attribute that can have no value for an instance [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.134]

3.2728

optional element

- 1.** element that can be present in a tag

3.2729

optional nonidentifying relationship

- 1.** nonidentifying relationship in which an instance of the child entity can exist without being related to an instance of the parent entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.135]
cf. mandatory nonidentifying relationship. nonidentifying relationship [key style]

3.2730

optional requirement

- 1.** requirement of a normative document that must be fulfilled in order to comply with a particular option permitted by that document [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1*, 3.6]

Note 1 to entry: An optional requirement is either: a) one of two or more alternative requirements, or b) an additional requirement that is fulfilled only if applicable and otherwise is disregarded.]

3.2731

optional task

1. those verification and validation (V&V) tasks that can be added to the minimum V&V tasks to address specific application requirements [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1]
2. task that can be added to the minimum testing tasks to address specific requirements.

3.2732

ORB

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

3.2733

ORB core

1. ORB component which moves a request from a client to the appropriate adapter for the target object [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.10]

3.2734

order clash

1. in software design, a type of structure clash in which a program must deal with two or more data sets that have been sorted in different orders

cf. data structure-centered design

3.2735

ordinal scale

1. scale in which the measurement values are rankings

cf. interval scale, nominal scale, ratio scale

Note 1 to entry: For example, the assignment of defects to a severity level is a ranking

3.2736

organization

1. group of people and facilities with an arrangement of responsibilities, authorities and relationships [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services*, 2.4; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.28; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.27]
2. people and processes assembled to produce a specific output (product or service) [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections*, 3.9]
3. person or a group of people and facilities with an arrangement of responsibilities, authorities and relationships to achieve its objectives [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.34]

EXAMPLE: company, corporation, firm, enterprise, institution, charity, sole trader, association, or parts or combination thereof

Note 1 to entry: An identified part of an organization (even as small as a single individual) or an identified group of organizations can be regarded as an organization if it has responsibilities, authorities and relationships. A body of persons organized for some specific purpose, such as a club, union, corporation, or society, is an organization. An organization can be public or private. The arrangement is generally orderly.

3.2737

organization chain

1. constellation of organizations that have business relationships with one another [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.25]

Note 1 to entry: The following are two types of chains identified: - supply chains of the IT organizations that are involved in the management and operation of the application (application manager, computer center, workspace manager, network manager, business information manager, suppliers, etc.); - business chains in which the user organization using the application participates (the business process supported by the application forms part of a chain across several organizations; for example, the chain of criminal justice, the healthcare chain).

3.2738

organization chart

1. graphical depiction of hierarchies and interrelationships among persons working together

3.2739

organization level

1. management level or levels responsible for managing one or more data processing or information systems organizations

3.2740

organizational breakdown structure (OBS)

1. A hierarchical representation of the project organization that illustrates the relationship between project activities and the organizational units that will perform those activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2741

organizational management profile

1. profile targeted at very small entities (VSEs) to provide them with additional organizational management guidance [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.35]

3.2742

organizational maturity

1. extent to which an organization has explicitly and consistently deployed processes that are documented, managed, measured, controlled, and continually improved.

Note 1 to entry: Organizational maturity can be measured via appraisals.

3.2743

organizational policy

1. guiding principle typically established by senior management that is adopted by an organization to influence and determine decisions

3.2744

organizational process assets

1. plans, processes, policies, procedures and knowledge bases, specific to and used by the performing organization [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 2. artifacts that relate to describing, implementing, and improving processes, such as policies, measurements, process descriptions, and process implementation support tools

cf. process asset library

Note 1 to entry: The term 'process assets' is used to indicate that these artifacts are developed or acquired to meet the business objectives of the organization and that they represent investments by the organization that are expected to provide current and future business value.

3.2745

organizational process maturity

1. extent to which an organizational unit consistently implements processes within a defined scope that contributes to the achievement of its business needs (current or projected) [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.2]

3.2746

organizational profile

1. set of process profiles [*ISO/IEC TR 29110-3-1:2015 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-1: Assessment Guide*, 3.3]

Note 1 to entry: Profiles conform to the organizational maturity levels that correspond to the basic, intermediate, and advanced profiles.

3.2747

organizational project management maturity

- 1.** the level of an organization's ability to deliver the desired strategic outcomes in a predictable, controllable and reliable manner [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2748

organizational test process

- 1.** test process for developing and managing organizational test specifications [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.23]

3.2749

organizational test specification

- 1.** document that provides information about testing for an organization, i.e. information that is not project-specific [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.24]

3.2750

organizational test strategy

- 1.** document that expresses the generic requirements for the testing to be performed on all the projects run within the organization, providing detail on how the testing is to be performed [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.25]

3.2751

organizational unit

- 1.** part of an organization that is the subject of measurement [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.30] **2.** identified part of an organization that deploys one or more processes that operate within a coherent set of business goals and which forms the basis for the scope of an assessment [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.2.14]

Note 1 to entry: An organizational unit is typically part of a larger organization, although in a small organization the organizational unit can be the whole organization.

3.2752

origin

- 1.** address of the initial storage location assigned to a computer program in main memory
cf. assembled origin, loaded origin, starting address

3.2753

origin attribute

- 1.** classification of software as either developed or nondeveloped

3.2754

original equipment manufacturer license

OEM license

- 1.** license for products or components that are created or manufactured by one company and licensed by another company [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.17]

3.2755

original source statement

- 1.** source statement that is obtained from an external product

3.2756

orphan page

- 1.** page on a website with no link from the home page or other page on the website [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.20]

3.2757

OSE

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

3.2758

OSF

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

3.2759

OSI

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

3.2760

OT

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

3.2761

OTE

1. operational test and evaluation [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.2762

OTP

1. one-time programming **2.** one-time password [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2763

OTRR

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

3.2764

outer cardinality

1. number of allowed instances of a participating data object from the viewpoint of the other participants in the relationship [*ISO/IEC 15476-4:2005 Information technology — CDIF semantic metamodel — Part 4: Data models, 6.6.1*]

cf. inner cardinality

3.2765

output

1. data transmitted to an external destination **2.** pertaining to a device, process, or channel involved in transmitting data to an external destination **3.** process by which an information processing system, or any of its parts, transfers data outside of that system or part [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **4.** a product, result, or service generated by a process. May be an input to a successor process [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **5.** in an IDEF0 model, that which is produced by a function [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.89*] **6.** pertaining to data transmitted to an external destination **7.** to transmit data to an external destination

3.2766

output arc (of a transition)

1. arc directed from the transition to a place [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.1.2*]

3.2767

output argument

- 1.** argument that has not been specified as an input argument [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.136*]
cf. input argument

Note 1 to entry: It is possible for an output argument to have no value at the time a request is made.

3.2768

output arrow

- 1.** arrow or arrow segment that expresses IDEF0 output [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.90*]

Note 1 to entry: That is, an object type set whose instances are created by a function by transforming the function's input. The arrowtail of an output arrow is attached to the right side of a box.

3.2769

output assertion

- 1.** logical expression specifying one or more conditions that program outputs must satisfy in order for the program to be correct
cf. input assertion, loop assertion, inductive assertion method

3.2770

output place (of a transition)

- 1.** place connected to the transition by an output arc [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.20.2*]

3.2771

output primitive

- 1.** primitive that includes source statements, function points, and documents

3.2772

output product

- 1.** the physical form that information can take and that an application distributes [*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*]

EXAMPLE: a report, an output file, or a message to a different application

3.2773

output sort

range sort

- 1.** sort of an output of an operator [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.23.3*]

3.2774

outsider's viewpoint

- 1.** perspective of a potential acquirer who does not own either an existing system nor its proposed replacement, and who evaluates the alternatives of buying the existing system at its salvage value or buying a replacement candidate system

Note 1 to entry: allows sunk cost and salvage values to be properly accounted for in a decision to replace a system

3.2775

overflow exception

- 1.** exception that occurs when the result of an arithmetic operation exceeds the size of the storage location designated to receive it

cf. addressing exception, data exception, operation exception, protection exception, underflow exception

3.2776

overhead time

- 1.** amount of time a computer system spends performing tasks that do not contribute directly to the progress of any user task

EXAMPLE: time spent tabulating computer resource usage for billing purposes

3.2777

overlay

- 1.** storage allocation technique in which computer program segments are loaded from auxiliary storage to main storage when needed, overwriting other segments not currently in use **2.** computer program segment that is maintained in auxiliary storage and loaded into main storage when needed, overwriting other segments not currently in use **3.** to load a computer program segment from auxiliary storage to main storage in such a way that other segments of the program are overwritten

3.2778

overlay supervisor

- 1.** routine that controls the sequencing and positioning of overlays

3.2779

overload

- 1.** to assign an operator, identifier, or literal more than one meaning, depending upon the data types associated with it at any given time during program execution

3.2780

override

- 1.** ability of a property in a subclass to respecify the realization of an inherited property of the same name while retaining the same meaning [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.137]

3.2781

overriding property

- 1.** property in a subclass that has the same meaning and signature as a similarly named property in one of its superclasses, but has a different realization [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.138]

3.2782

owned attribute

- 1.** attribute of an entity that has not migrated into the entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.139]

Note 1 to entry: [key style]

3.2783

owner

- 1.** person or organization that owns the copyright for the Candidate FSM method [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1*, 3.7]

3.2784

owner of the FSM method

- 1.** the person or organization that owns the intellectual property rights for the FSM method [*ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*, 3.7]

3.2785

P&D

- 1.** production and deployment [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs*, 3.2]

3.2786

P-V pair

1. combination of a test item parameter with a value assigned to that parameter, used as a test condition and coverage item in combinatorial test design techniques [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.22]

3.2787

P/T net

1. Place/Transition net [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.2.4]

3.2788

PA

1. process attribute [*ISO/IEC TR 29110-3-1:2015 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-1: Assessment Guide*, 4.2]

3.2789

pack

1. to store data in a compact form in a storage medium, using known characteristics of the data and medium in such a way as to permit recovery of the data

cf. unpack

3.2790

package

1. separately compilable software component consisting of related data types, data objects, and subprograms **2.** set of related components that are combined into a single distributable item **3.** namespace for the grouped elements [*ISO/IEC 30130:2016 Software engineering — Capabilities of software testing tools*] *cf.* data abstraction, encapsulation, information hiding

EXAMPLE: a software package having a set of files that can be used to install software on a computing device and can be distributed via CD or electronic means

3.2791

packaging

1. in software development, the assignment of modules to segments to be handled as distinct physical units for execution by a computer

3.2792

padding

1. technique of filling out a fixed-length block of data with dummy characters, words, or records **2.** dummy characters, words, or records used to fill out a fixed-length block of data

3.2793

page

1. fixed-length segment of data or of a computer program treated as a unit in storage allocation **2.** in a virtual storage system, a fixed-length segment of data or of a computer program that has a virtual address and is transferred as a unit between main and auxiliary storage **3.** screenful of information on a video display terminal **4.** structuring mechanism used to split a large net graph into smaller parts, which are also the units of the net to be printed [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.8] *cf.* paging

3.2794

page breakage

1. portion of main storage that is unused when the last page of data or of a computer program does not fill the entire block of storage allocated to it

cf. paging

3.2795

page frame

1. block of main storage having the size of, and used to hold, a page

cf. paging

3.2796

page reference

1. expression that unambiguously identifies a model page [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.91]

Note 1 to entry: The page reference incorporates a diagram reference to the associated diagram, the type of page, and any sequencing data needed to distinguish different pages of the same type that are associated with the same diagram.

3.2797

page swapping

1. exchange of pages between main storage and auxiliary storage

cf. paging

3.2798

page table

1. table that identifies the location of pages in storage and gives significant attributes of those pages

cf. paging

3.2799

page type letter

1. uppercase letter in a page reference that denotes a specific type of model page [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.92]

3.2800

page zero

1. in the paging method of storage allocation, the first page in a series of pages

3.2801

pager

1. routine that initiates and controls the transfer of pages between main and auxiliary storage

cf. paging

3.2802

paging

block allocation

1. storage allocation technique in which programs or data are divided into fixed-length blocks called pages, main storage is divided into blocks of the same length called page frames, and pages are stored in page frames, not necessarily contiguously or in logical order **2.** storage allocation technique in which programs or data are divided

into fixed-length blocks called pages, main storage is divided into blocks of the same length called page frames, and pages are transferred between main and auxiliary storage as needed **3.** transfer of pages as in (2)

cf. contiguous allocation, page, page breakage, page frame, page swapping, page table, page zero, pager, working set

3.2803

PAM

1. process assessment model [*ISO/IEC TR 29110-3-1:2015 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-1: Assessment Guide*, 4.2]

3.2804

parallel

1. pertaining to the simultaneous transfer, occurrence, or processing of the individual parts of a whole, such as the bits of a character, using separate facilities for the various parts

cf. serial (1), concurrent

3.2805

parallel classes

1. pair of classes that are distinct, are not mutually exclusive and have a common generic ancestor class and for which neither is a generic ancestor of the other [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.140*]

3.2806

parallel construct

1. program construct consisting of two or more procedures that can occur simultaneously

3.2807

parallel run operation

1. operation of two information processing systems, a given one and its intended replacement, with the same application and source data, for comparison and confidence [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2808

parameter

1. variable that is given a constant value for a specified application **2.** constant, variable, or expression that is used to pass values between software modules **3.** symbol that can take a range of values defined by a set it is defined as a constant in the signature [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.18*]

cf. adaptation

3.2809

parameterized collection class

1. collection class restricted to hold only instances of a specified type (class) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.141*]

3.2810

parameterized high-level net graph

1. high-level net graph that contains parameters in its definition [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.19*]

3.2811

parametric estimating

1. estimating technique in which an algorithm is used to calculate cost or duration based on historical data and project parameters [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2812

parent box

1. ancestral box related to its child diagram by exactly one parent/child relationship [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.93*]

Note 1 to entry: That is, a box detailed by a child diagram. The existence of this child diagram is indicated by a box detail reference.

3.2813

parent diagram

1. diagram that contains a parent box [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.94*]

3.2814

parent entity

1. entity in a specific relationship whose instances can be related to a number of instances of another entity (child entity) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.142*]

Note 1 to entry: [key style]

3.2815

parent function

- 1.** function modeled by a parent box [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.95]

3.2816

Pareto diagram

Pareto chart

- 1.** histogram, ordered by frequency of occurrence, that shows how many results were generated by each identified cause [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2817

parking lot diagram

- 1.** displayed listing of incomplete tasks or user stories not yet being worked or completed. This listing can be grouped by function, with the estimated priority and expected date to start, finish, or dispose of the items [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.2818

parse

- 1.** to determine the syntactic structure of a language unit by decomposing it into more elementary subunits and establishing the relationships among the subunits

EXAMPLE: to decompose blocks into statements, statements into expressions, expressions into operators and operands

3.2819

parser

- 1.** software tool that parses computer programs or other text, often as the first step of assembly, compilation, interpretation, or analysis

3.2820

partial

- 1.** incomplete mapping [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.143]

cf. total, mapping completeness, optional

Note 1 to entry: That is, some instances map to no related instance. An attribute can be declared partial, meaning it may have no value. A participant property is declared optional as part of the relationship syntax. An operation is declared partial when it has no meaning for some instances, i.e., it does not give an answer or produce a response.

3.2821

partial cluster

incomplete cluster

- 1.** subclass cluster in which an instance of the superclass exists without also being an instance of any of the subclasses [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.144]

cf. total cluster, superclass

3.2822

partial correctness

- 1.** in proof of correctness, a designation indicating that a program's output assertions follow logically from its input assertions and processing steps

cf. total correctness

3.2823

participant property

- 1.** property of a state class that reflects that class' knowledge of a relationship in which instances of the class participate [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.145]

Note 1 to entry: When a relationship exists between two state classes, each class contains a participant property for that relationship. A participant property is a mapping from a state class to a related (not necessarily distinct) state class. The name

of each participant property is the name of the role that the other class plays in the relationship, or it simply the name of the class at the other end of the relationship (as long as using the class name does not cause ambiguity). A value of a participant property is the identity of a related instance.

3.2824

partitioning

- 1.** decomposition; the separation of the whole into its parts

3.2825

party

- 1.** organization entering into an agreement [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.29; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.28]
- 2.** enterprise object modeling a natural person or any other entity considered to have some of the rights, powers and duties of a natural person [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language*, 6.6.1]

EXAMPLE: enterprise objects representing natural persons, legal entities, governments and their parts, and other associations or groups of natural persons

Note 1 to entry: Parties are responsible for their actions and the actions of their agents. Parties to an agreement are called the acquirer and the supplier.

3.2826

pass

- 1.** single cycle in the processing of a set of data, usually performing part of an overall process

EXAMPLE: a pass of an assembler through a source program, a pass of a sort program through a set of data

3.2827

pass/fail criteria

- 1.** decision rules used to determine whether a software item or a software feature passes or fails a test [*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*, 4.1.10]
- 2.** decision rules used to determine whether a test item, or feature of a test item, has passed or failed after testing [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.26]

3.2828

passive I/O device

- 1.** device that does not generate an interrupt on completion of an input or output operation

Note 1 to entry: The input from a passive input device must be read either on a polled basis or on demand.

3.2829

passive I/O device interface task

- 1.** task that interfaces to a passive I/O device and either reads from it or writes to it on demand

3.2830

passive interconnection

- 1.** interoperability agreement describing a common interpretation of one or more phenomena shared between two interacting things [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections*, 3.10]

3.2831

passive object

- 1.** object with no thread of control

Note 1 to entry: an object with operations that concurrent objects (that is, tasks) invoke directly or indirectly

3.2832

patch

- 1.** modification made directly to an object program without reassembling or recompiling from the source program
- 2.** software component that, when installed, directly modifies files or device settings related to a different software component without changing the version number or release details for the related software component
[ISO/IEC 19770-2:2015 Information technology — Software asset management — Part 2: Software identification tag, 3.1.1]
- 3.** modification to a source or object program
- 4.** to perform a modification as in (1), (2), or (3)

3.2833

path

- 1.** in software engineering, a sequence of instructions that are performed in the execution of a computer program
- 2.** in file access, a hierarchical sequence of directory and subdirectory names specifying the storage location of a file,
- 3.** sequence of executable statements of a test item
[ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.23]

3.2834

path analysis

- 1.** analysis of a computer program to identify all possible paths through the program, to detect incomplete paths, or to discover portions of the program that are not on any path

3.2835

path condition

- 1.** set of conditions that must be met in order for a particular program path to be executed

3.2836

path convergence

- 1.** a relationship in which a schedule activity has more than one predecessor
[A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.2837

path divergence

- 1.** a relationship in which a schedule activity has more than one successor
[A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.2838

path expression

- 1.** logical expression indicating the input conditions that must be met in order for a particular program path to be executed

3.2839

path testing

- 1.** testing designed to execute all or selected paths through a computer program
cf. branch testing, statement testing

3.2840

pathological coupling

- 1.** type of coupling in which one software module affects or depends upon the internal implementation of another
cf. common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling

3.2841

pause

- 1.** to suspend the execution of a computer program
cf. halt, stop

3.2842

payment system

- 1.** the system used to provide and track suppliers' invoices and payments for services and products
[A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.2843

payoff matrix

1. in decisions under uncertainty, a table relating the desirability of a set of alternatives to a set of future states

3.2844

PBO

1. project-based organization [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2845

PCA

1. physical configuration audit **2.** programmable counter array

3.2846

PCB

1. printed circuit board

3.2847

PCBA

1. printed circuit board assembly

3.2848

PCO

1. Point of Control and Observation [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*]

3.2849

PDCA

1. Plan-Do-Check-Act

3.2850

PDL

1. program design language

3.2851

PDM

1. precedence diagramming method [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2852

PDPC

1. process decision program chart [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2853

PDR

1. preliminary design review [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.2854

peer review

1. review of work products performed by others qualified to do the same work
cf. inspection, structured walkthrough, work product

Note 1 to entry: often performed during development of the work products to identify defects for removal. The intent is to increase the quality of the work product as well as to reduce cost by fixing defects as soon as possible.

3.2855

peer software

1. piece of software that resides in the same layer as, and exchanges data with, another piece of software [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.21*]

3.2856

PEO

1. program executive office [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2857

percent complete

1. an estimate, expressed as a percent, of the amount of work that has been completed on an activity or a work breakdown structure component [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2858

perceptual reference point

1. reference point at which there is some interaction between the system and the physical world [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 15.3.2]

3.2859

perfective maintenance

1. modification of a software product after delivery to detect and correct latent faults in the software product before they are manifested as failures [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance*, 3.7; *ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.42] **2.** software maintenance performed to improve the performance, maintainability, or other attributes of a computer program **3.** improvements in software's performance or functionality, for example, in response to user suggestions and requests
cf. adaptive maintenance, corrective maintenance

Note 1 to entry: Perfective maintenance provides enhancements for users, improvement of program documentation, and recoding to improve software performance, maintainability, or other software attributes.

3.2860

perform integrated change control

1. the process of reviewing all change requests, approving changes, and managing changes to deliverables, organizational process assets, project documents, and the project management plan, and communicating their disposition [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2861

perform qualitative risk analysis

1. the process of prioritizing risks for further analysis or action by assessing and combining their probability of occurrence and impact [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2862

perform quality assurance

1. the process of auditing the quality requirements and the results from quality control measurements to ensure that appropriate quality standards and operational definitions are used [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2863

perform quantitative risk analysis

1. the process of numerically analyzing the effect of identified risks on overall project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2864

performance

1. degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage **2.** extent to which the execution of an application in the production environment achieves its purpose in terms of speed of input, transfer, processing, storage and output (the response speed of an application observed by an end user) [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.26]

3.2865

performance analysis

- 1.** quantitative analysis of a real-time system (or software design) executing on a given hardware configuration with a given external workload applied to it

3.2866

performance baseline

- 1.** result from a normal execution of a performance workload against a system without performing disturbance injection [*ISO/IEC 25045:2010 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability*, 4.1]

3.2867

performance deficiency

- 1.** difference between the required (or desired) level of performance and the actual performance [*ISO/IEC 25064:2013 Systems and software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: User needs report*, 4.9]

Note 1 to entry: Performance deficiencies can include deficiencies in measured customer satisfaction. Deficiency data is obtainable only in environments where specific performance requirements exist.

3.2868

performance efficiency

- 1.** performance relative to the amount of resources used under stated conditions [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.2]

Note 1 to entry: Resources can include other software products, the software and hardware configuration of the system, and materials (e.g. print paper, storage media).

3.2869

performance measurement baseline

PMB

- 1.** an approved integrated scope-schedule-cost plan for the project work against which project execution is compared to measure and manage performance. The PMB includes contingency reserve, but excludes management reserve [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition]

3.2870

performance requirement

- 1.** measurable criterion that identifies a quality attribute of a function or how well a functional requirement shall be accomplished [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2] **2.** system or software requirement specifying a performance characteristic that a system/software system or system/software component must possess **3.** requirement that imposes conditions on a functional requirement.
cf. nonfunctional requirement

EXAMPLE: a requirement that specifies the speed, accuracy, or memory usage with which a given function must be performed

Note 1 to entry: A performance requirement is an attribute of a functional requirement.

3.2871

performance reviews

- 1.** a technique that is used to measure, compare, and analyze actual performance of work in progress on the project against the baseline [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition]

3.2872

performance specification

- 1.** document that specifies the performance characteristics that a system or component must possess

Note 1 to entry: often part of a requirements specification. These characteristics typically include speed, accuracy, and memory usage.

3.2873

performance testing

1. type of testing conducted to evaluate the degree to which a test item accomplishes its designated functions within given constraints of time and other resources [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.27]

cf. functional testing

3.2874

performed process

1. process that accomplishes the needed work to produce work products

Note 1 to entry: satisfies the specific goals of the process area

3.2875

performing organization

1. an enterprise whose personnel are most directly involved in doing the work of the project or program [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2876

periodic I/O device interface task

1. task that interfaces to a passive I/O device and polls it regularly

3.2877

periodic task

1. task that a timer event activates at regular intervals

3.2878

peripheral equipment

1. device that is controlled by and can communicate with a particular computer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: external storage

3.2879

permanence

1. degree to which failures can affect object state changes due to completed transactions [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 13.7.1.5]

3.2880

permission

1. prescription that a particular behavior is allowed to occur [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 11.2.5]

3.2881

perpetual license

1. license for a software entitlement granted in perpetuity [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.18]

Note 1 to entry: The alternative to a perpetual license is a term or subscription-based license.

3.2882

persistence

1. property that an object continues to exist across changes of contractual context or of epoch [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 11.3.1]

3.2883

persistence schema

1. specification of constraints on the use of specific processing, storage and communication functions [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 16.5.1.1]

3.2884

persistence transparency

- 1.** distribution transparency which masks, from an object, the deactivation and reactivation of other objects (or itself) [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.4.1.7]

Note 1 to entry: Deactivation and reactivation are often used to maintain the persistence of an object when a system is unable to provide it with processing, storage and communication functions continuously.

3.2885

persistent

- 1.** for a URL, describing a reference that does not need to change at the link in a document, and can still reach the desired object even though that object has changed locations [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.21]

3.2886

persistent storage

- 1.** storage which enables a functional process to store data beyond the life of the functional process, or which enables a functional process to retrieve data stored by another functional process, or stored by an earlier occurrence of the same functional process, or stored by some other process [ISO/IEC 19761:2011 *Software engineering — COSMIC: a functional size measurement method*, 2.22]

Note 1 to entry: As persistent storage is on the software side of the boundary; it is not considered to be a functional user of the software being measured.

3.2887

persona

- 1.** archetypical user of a system, based on research into real users of a system [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.8] **2.** representation of a type of user that includes a concise summary of the characteristics of the user that is most informative to the design or illustrative of specific user requirements [ISO/IEC 25063:2014 *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description*]

Note 1 to entry: A persona typically includes behavior patterns, goals, skills, attitudes, and environment, with a few fictional personal details to make the persona a realistic character.

3.2888

personal computer (PC)

- 1.** microcomputer primarily intended for stand-alone use by an individual [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.2889

personnel

- 1.** individual expected to perform duties on behalf of the organization, including officers, employees, and contractors [ISO/IEC 19770-1:2012 *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3. 8]

3.2890

personnel management

- 1.** management of activities involving hiring, retaining, promoting, training, and terminating personnel

3.2891

PERT

- 1.** program evaluation review technique [ISO/IEC/IEEE 16326:2009 *Systems and software engineering — Life cycle processes — Project management*, 3]

3.2892

pessimistic duration

- 1.** estimate of the longest activity duration that takes into account all of the known variables that could affect performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2893

PESTEL

- 1.** political, economic, social, technological, environmental, and legal

3.2894

Petri net

- 1.** algebraic structure with two sets, one called places and the other called transitions, together with their associated relations and functions, and named after their inventor, Carl Adam Petri [ISO/IEC 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.16.3] **2.** abstract, formal model of information flow, showing static and dynamic properties of a system

Note 1 to entry: A Petri net is usually represented as a graph having two types of nodes (called places and transitions) connected by arcs, and markings (called tokens) indicating dynamic properties.

3.2895

PG

- 1.** profile group [ISO/IEC TR 29110-1:2016 *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 4.2]

3.2896

phase gate

- 1.** a review at the end of a phase in which a decision is made to continue to the next phase, to continue with modification, or to end a project [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.2897

physical configuration audit (PCA)

- 1.** audit conducted to verify that a configuration item, as built, conforms to the technical documentation that defines it [IEEE 828-2012 *IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] cf. functional configuration audit

Note 1 to entry: For software, the purpose of the software physical configuration audit (PCA) is to ensure that the design and reference documentation is consistent with the as-built software product.

3.2898

physical requirement

- 1.** requirement that specifies a physical characteristic that a system or system component must possess cf. design requirement, functional requirement, implementation requirement, interface requirement, performance requirement

EXAMPLE: material, shape, size, weight

3.2899

physical source statement (PSS)

- 1.** source statement considered as a line of code
cf. logical source statement

3.2900

Pla

- 1.** provide instruments activity [ISO/IEC 29155-2:2013: *Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*, 4]

3.2901

PICS

- 1.** Protocol Implementation Conformance Statement [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 4]

3.2902

picture

- 1.** illustration that shows the actual appearance of physical objects [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.32]

EXAMPLE: photograph, drawing

3.2903

PII

1. personally identifiable information [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2904

pilot project

1. project designed to test a preliminary version of an information processing system under actual but limited operating conditions and which will then be used to test the definitive version of the system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2905

PIM

1. platform independent model [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.2906

PIN

1. personal identification number [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.2907

pipeline

1. software or hardware design technique in which the output of one process serves as input to a second, the output of the second process serves as input to a third, and so on, often with simultaneity within a single cycle time

3.2908

pixel

1. smallest element of a screen display; short for 'picture element'

3.2909

PIXIT

1. Protocol Implementation Extra Information for Testing [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 4*]

3.2910

place

1. node of a net, taken from the place kind, normally represented by an ellipse in the net graph [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.20*]

Note 1 to entry: A place is typed

3.2911

place type

1. non-empty set of data items associated with a place [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.20.3*]

Note 1 to entry: This set can describe an arbitrarily complex data structure.

3.2912

place/transition net

1. Petri Net comprising a net graph with positive integers associated with arcs and an initial marking function which associates a natural number of simple tokens ('black dots') with places [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.16.4*]

3.2913

plan

1. information item that presents a systematic course of action for achieving a declared purpose, including when, how, and by whom specific activities are to be performed [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.16]

3.2914

plan communications management

1. the process of developing an appropriate approach and plan for project communications based on stakeholder's information needs and requirements, and available organizational assets [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2915

plan cost management

1. the process that establishes the policies, procedures, and documentation for planning, managing, expending, and controlling project costs [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2916

plan human resource management

1. the process of identifying and documenting project roles, responsibilities, required skills, reporting relationships, and creating a staffing management plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2917

plan procurement management

1. the process of documenting project procurement decisions, specifying the approach, and identifying potential sellers [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2918

plan quality management

1. the process of identifying quality requirements and/or standards for the project and its deliverables, and documenting how the project will demonstrate compliance with quality requirements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2919

plan risk management

1. the process of defining how to conduct risk management activities for a project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2920

plan risk responses

1. the process of developing options and actions to enhance opportunities and to reduce threats to project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2921

plan schedule management

1. the process of establishing the policies, procedures, and documentation for planning, developing, managing, executing, and controlling the project schedule [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2922

plan scope management

1. the process of creating a scope management plan that documents how the project scope will be defined, validated, and controlled [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2923

plan stakeholder management

1. the process of developing appropriate management strategies to effectively engage stakeholders throughout the project life cycle, based on the analysis of their needs, interests, and potential impact on project success [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2924

plan standard

- 1.** standard that describes the characteristics of a scheme for accomplishing defined objectives or work within specified resources

3.2925

planned process

- 1.** process that is documented by both a description and a plan

Note 1 to entry: The related process description and plan are coordinated, and the plan can include standards, requirements, objectives, resources, and assignments.

3.2926

planned value (PV)

budgeted cost of work scheduled (BCWS)

- 1.** the authorized budget assigned to scheduled work [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2927

planning horizon

- 1.** consistent time span used to compare the cost of alternatives

3.2928

planning package

- 1.** a work breakdown structure component below the control account with known work content but without detailed schedule activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. control account

3.2929

planning process group

- 1.** those processes required to establish the scope of the project, refine the objectives, and define the course of action required to attain the objectives that the project was undertaken to achieve [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2930

plastic leaded chip carrier (PLCC)

- 1.** rectangular chip unit, made of plastic for use in low-heat devices, usually with surface-mount or J-shaped (J-lead) connectors

3.2931

platform

- 1.** type of computer or hardware device and/or associated operating system, or a virtual environment, on which software can be installed or run [*ISO/IEC 19770-1:2012 Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3.9] **2.** a collection of hardware and software components that are needed for a CASE tool to operate [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections*, 3.11] **3.** combination of an operating system and hardware that makes up the operating environment in which a program runs [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.26]
cf. device

Note 1 to entry: A platform is distinct from the unique instances of that platform, which are typically referred to as devices or instances.

3.2932

platform provider

- 1.** organization responsible for the platform [*ISO/IEC 19770-2:2015 Information technology — Software asset management — Part 2: Software identification tag*, 4.1.18]

Note 1 to entry: The platform provider is typically the vendor of the relevant operating system or virtual environment.

3.2933

PLCC

1. plastic leaded chip carrier

3.2934

pleasure

1. degree to which a user obtains pleasure from fulfilling personal needs [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.3.3]

Note 1 to entry: Personal needs can include needs to acquire new knowledge and skills, to communicate personal identity and to provoke pleasant memories.

3.2935

plurality

1. decisions made by the largest block in a group, even if a majority is not achieved [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2936

PM

1. Project Manager [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 5] **2.** program manager

3.2937

PM&P

1. parts, materials, and processes [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2938

PMB

1. performance measurement baseline [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2939

PMBOK®

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

3.2940

PMI

1. Project Management Institute [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*]

3.2941

PMIS

1. project management information system [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2942

PMO

1. project management office**2.** program management office [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2943

PMP

1. Project Management Plan [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 3] **2.** Program Management Plan

3.2944

PMP®

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

3.2945

PN

- 1.** Petri Net [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.2.5]

3.2946

PNG

- 1.** Portable Network Graphics [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.2947

PNML

- 1.** Petri Net Markup Language [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.2.5]

3.2948

PNML Core Model

- 1.** metamodel defining the basic concepts and structure of net graph models that are common to all versions of Petri nets [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.9]

3.2949

PNML document

Petri Net document

- 1.** XML document that contains one or more net graphs [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.10]

3.2950

PNML high-level net document

- 1.** PNML Document that contains one or more net graphs, where all net graphs conform to high-level Petri nets [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.11]

3.2951

PNML place/transition net document

- 1.** PNML document that contains one or more net graphs, where all net graphs conform to place/transition nets [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 4.1.12]

3.2952

PNML symmetric net document

- 1.** PNML Document that contains one or more net graphs, where all net graphs conform to symmetric nets [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*]

3.2953

PO

- 1.** purchase order [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile*, 4.2]

3.2954

POA

- 1.** portable object adapter [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.2955

point

- 1.** measure of vertical distance; there are approximately 28 points to the millimeter (72 points to the inch)

3.2956

point design

- 1.** selection of one design that satisfies the requirements without examining other, potentially more effective, designs

3.2957

pointer

- 1.** data item that specifies the location of another data item

EXAMPLE: a data item that specifies the address of the next employee record to be processed

3.2958

policy

- 1.** set of rules related to a particular purpose **2.** clear and measurable statements of preferred direction and behavior to condition the decisions made within an organization *[ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.17]* **3.** constraint on a system specification foreseen at design time, but whose detail is determined subsequent to the original design, and capable of being modified from time to time in order to manage the system in changing circumstances *[ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.8]* **4.** a structured pattern of actions adopted by an organization such that the organization's policy can be explained as a set of basic principles that govern the organization's conduct *[A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]*

Note 1 to entry: A rule can be expressed as an obligation, an authorization, a permission, or a prohibition. Not every policy is a constraint. Some policies represent an empowerment.

3.2959

policy declaration

- 1.** element in a specification defined in order to allow incorporation of future constraints, together with rules determining the allowed form of acceptable constraints and the circumstances in which such constraints can be applied *[ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.9]*

3.2960

policy envelope

- 1.** set of acceptable policy values that could be applied at a particular policy declaration *[ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.11]*

3.2961

policy value

- 1.** specific constraints associated with a policy in some particular epoch *[ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.10]*

3.2962

policy-setting behavior

- 1.** behavior defined in a specification via which a policy can be changed *[ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.12]*

3.2963

pop-up

- 1.** embedded, context-sensitive information that is displayed when invoked by user action *[ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation, 3.27]*

3.2964

port

- 1.** surface feature through which clients and other elements of an application environment can interact with a component *[ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1]* **2.** interface on a unit's boundary that allows a subset of the interactions that can be exchanged with another unit *[IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.19]*

Note 1 to entry: The component model supports four basic kinds of ports: facets, receptacles, event sources, event sinks and attributes.

3.2965

port alias

1. replacement relationship in a build specification that identifies a port of one unit with a port of a sub-unit and indicates that interactions at the two ports can be paired identically or compatibly [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.20]

Note 1 to entry: A port alias is used to indicate identity or compatibility of two ports at different levels of assembly in which the port in a lower level unit of assembly serves as a realization of a port at a higher -level unit of assembly.

3.2966

port couple

1. interconnection relationship in a build specification that identifies a port of one unit with a port of another (structural peer) unit and indicates that interactions output from one are input to the other, and vice versa [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.21]

Note 1 to entry: A port couple defines a one-to-one matching of the two ports defining opposite sides of a particular interface.

3.2967

port-to-port time

1. elapsed time between the application of a stimulus to an input interface and the appearance of the response at an output interface

cf. response time, think time, turnaround time

3.2968

portability

transportability

1. ease with which a system or component can be transferred from one hardware or software environment to another **2.** capability of a program to be executed on various types of data processing systems without converting the program to a different language and with little or no modification [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3. degree of effectiveness and efficiency with which a system, product, or component can be transferred from one hardware, software or other operational or usage environment to another [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.8/ **4.** property that the reference points of an object allow it to be adapted to a variety of configurations [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 15.4.1]

cf. machine-independent

3.2969

portability testing

1. type of testing conducted to evaluate the ease with which a test item can be transferred from one hardware or software environment to another, including the level of modification needed for it to be executed in various types of environments [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.28]

3.2970

portable computer

1. microcomputer that can be hand-carried for use in more than one location [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.2971

portfolio

1. projects, programs, subportfolios, and operations managed as a group to achieve strategic objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2972

portfolio management

1. the centralized management of one or more portfolios to achieve strategic objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2973

post-closure activities

- 1.** activities that occur after a software system has been formally accepted by its customer

Note 1 to entry: These activities include, but are not limited to, lessons-learned reviews and archiving project materials.

3.2974

postcondition

- 1.** condition that is guaranteed to be true after a successful property request [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.147] **2.** constraint that must be true when a use case has ended **3.** predicate that a specification requires to be true immediately after the occurrence of an action [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.30]

3.2975

postmortem dump

- 1.** dump that is produced upon abnormal termination of a computer program

cf. change dump, dynamic dump, memory dump, selective dump, snapshot dump, static dump

3.2976

postprocessor

- 1.** computer program or routine that carries out some final processing step after the completion of the primary process

cf. preprocessor

EXAMPLE: a routine that reformats data for output

3.2977

power-down mode

- 1.** energy-saving operational state for a microcontroller unit (MCU)

3.2978

powertype

- 1.** type, the instances of which are subtypes of another type called the 'partitioned type' [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies*, 3.12]

EXAMPLE: The class TreeSpecies is a powertype of the class Tree, since each instance of TreeSpecies is also a subclass of Tree.

3.2979

PPL

- 1.** Product Parts List [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 3]

3.2980

PPP

- 1.** program protection plan [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2981

PPSL

- 1.** program parts selection list [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.2982

PR&RPI

- 1.** Problem Reporting and Resolution Planned Information.

3.2983

practice

1. specific type of activity that contributes to the execution of a process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.3.8] 2. a specific type of professional or management activity that contributes to the execution of a process and that can employ one or more techniques and tools [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] cf. conventions, standards

3.2984

precedence diagramming method (PDM)

activity-on-node (AON)

1. a technique used for constructing a schedule model in which activities are represented by nodes and are graphically linked by one or more logical relationships to show the sequence in which the activities are to be performed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2985

precedence relationship

1. the term used in the precedence diagramming method for a logical relationship. In current usage, however, precedence relationship, logical relationship, and dependency are widely used interchangeably, regardless of the diagramming method used [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. logical relationship

3.2986

precision

1. degree of exactness or discrimination with which a quantity is stated 2. Within the quality management system, precision is a measure of exactness [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. accuracy

EXAMPLE: a precision of 2 decimal places versus a precision of 5 decimal places

3.2987

precompiler

1. computer program or routine that processes source code and generates equivalent code that is acceptable to a compiler

cf. preprocessor

Note 1 to entry: for example, a routine that converts structured FORTRAN to ANSI-standard FORTRAN

3.2988

precondition

1. condition that is required to be true before making a property request [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.148] 2. constraint that must be true when a use case is invoked 3. predicate that a specification requires to be true for an action to occur [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*, 9.29]

3.2989

predecessor activity

1. an activity that logically comes before a dependent activity in a schedule [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2990

predicate

1. logical expression which evaluates to TRUE or FALSE, normally to direct the execution path in code [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.24]

3.2991

predicate data use

p-use

- 1.** data use associated with the decision outcome of the predicate portion of a decision statement [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.25*]

3.2992

predictive life cycle

- 1.** a form of project life cycle in which the project scope, and the time and cost required to deliver that scope, are determined as early in the life cycle as possible [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.2993

predictive metric

- 1.** metric applied during development and used to predict the values of a software quality factor [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.13*]

3.2994

predictive metric value

- 1.** numerical target related to a quality factor to be met during system development [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.14*]

Note 1 to entry: This is an intermediate requirement that is an early indicator of final system performance. For example, design or code errors can be early predictors of final system reliability.

3.2995

preliminary design

- 1.** process of analyzing design alternatives and defining the architecture, components, interfaces, and timing and sizing estimates for a system or component **2.** result of the process in (1)

cf. detailed design

3.2996

preliminary design review (PDR)

- 1.** review conducted to evaluate the progress, technical adequacy, and risk resolution of the selected design approach for one or more configuration items; to determine each design's compatibility with the requirements for the configuration item; to evaluate the degree of definition and assess the technical risk associated with the selected manufacturing methods and processes; to establish the existence and compatibility of the physical and functional interfaces among the configuration items and other items of equipment, facilities, software and personnel; and, as applicable, to evaluate the preliminary operational and support documents **2.** review as in (1) of any hardware or software component
cf. critical design review, system design review

3.2997

preparation time

- 1.** time which elapses before the task submission [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.12*]
cf. task mode

Note 1 to entry: The event of starting the preparation time depends on the definition of the task mode of the following task.

3.2998

preprocessor

- 1.** computer program or routine that carries out some processing step prior to the primary process
cf. postprocessor

EXAMPLE: a precompiler or other routine that reformats code or data for processing

3.2999

prescription

- 1.** action that establishes a rule [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.6.3*]

3.3000

present worth

- 1.** representation of a cash flow as a single instance at the beginning of the planning horizon
cf. future worth, annual equivalent

3.3001

presentable

- 1.** can be retrieved and viewed [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.18]

3.3002

presentation device

- 1.** device used to present data to the intended user of a system [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.30]

3.3003

prestore

- 1.** to store data that are required by a computer program or routine before the program or routine is entered

3.3004

prettyprinting

- 1.** use of indentation, blank lines, and other visual cues to show the logical structure of a program

3.3005

preventive action

- 1.** an intentional activity that ensures the future performance of the project work is aligned with the project management plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** action to avoid or eliminate the causes or reduce the likelihood of occurrence of a potential nonconformity or other potential undesirable situation

3.3006

preventive maintenance

- 1.** modification of a software product after delivery to detect and correct latent faults in the software product before they become operational faults [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance*, 3.8] **2.** maintenance performed for the purpose of preventing problems before they occur **3.** designing a software system that is easy to maintain **4.** continuously upgrading a system to enable it to cope with current and future changes

3.3007

previously developed software

- 1.** software that has been produced prior to or independent of the project for which the plan is prepared, including software that is obtained or purchased from outside sources [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans*, 3.1.2]

3.3008

primary Ent

primary entitlement schema

- 1.** entitlement schema (Ent) which encapsulates basic information about an entitlement [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.19]

Note 1 to entry: Primary Ents have an <entType> of Initial, Consolidation, AllocationReceived, or TransferReceived. These are base Ents which allow for initial population of an Ent into an Ent library (with the exception of Consolidation, which can be used to replace several previous Ents if desired). Other Ents (called supplemental Ents) can extend the data in these base type Ents.

3.3009

primary entity-type

- 1.** in Mk II FPA, one of the main entity-types which has the attributes that the application has been designed to process and/or store [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10]

3.3010

primary intent

- 1.** intent that is first in importance [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.43]

3.3011

primary key

- 1.** candidate key selected as the unique identifier of an entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.149] **2.** value that uniquely identifies component instances within the scope of the home that manages them [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

Note 1 to entry: [key style]

3.3012

primitive

- 1.** lowest level for which data is collected

EXAMPLE: error, failure, fault, time, time interval, date, number of non-commentary source code statements, edges, and nodes

Note 1 to entry: Primitives are directly measurable or countable, or can be given a constant value or condition for a specific measure.

3.3013

principal

- 1.** party that has delegated (authority, a function, etc.) to another [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language*, 6.6.9]

3.3014

printed circuit board (PCB)

- 1.** configuration of connections between electronic components, typically using metallic paths etched on a non-conducting substrate (board)

3.3015

printed circuit board assembly (PCBA)

- 1.** printed circuit board with embedded components and devices

3.3016

printed documentation

- 1.** documentation that is either provided in printed form, or provided in electronic form for the customer or user to print [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.33]

cf. embedded documentation

3.3017

prioritization matrices

- 1.** a quality management planning tool used to identify key issues and evaluate suitable alternatives to define a set of implementation priorities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3018

priority ceiling protocol

- 1.** algorithm that provides bounded priority inversion

Note 1 to entry: that is, at most one lower-priority task can block a higher-priority task

3.3019

priority interrupt

- 1.** interrupt performed to permit execution of a process that has a higher priority than the process currently executing

3.3020

priority inversion

1. case where a task's execution is delayed because a lower priority task is blocking it

3.3021

private

1. responsibility that is visible only to the class or the receiving instance of the class (available only within methods of the class) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.150] **2.** known only within a single routine or module
cf. protected, public, hidden

3.3022

private type

1. data type whose structure and possible values are defined but are not revealed to the user of the type
cf. information hiding

3.3023

privileged instruction

1. computer instruction that can be executed only by a supervisory program

3.3024

PRM

1. process reference model [*ISO/IEC TR 29110-3-1:2015 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-1: Assessment Guide*, 4.2]

3.3025

probability

1. extent to which an event is likely to occur [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management*, 3.3] **2.** mathematically, a real number in the scale 0 to 1 attached to a random event, related to a long-run relative frequency of occurrence or to a degree of belief that an event will occur [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management*, 3.3]

Note 1 to entry: For a high degree of belief, the probability is near 1. Frequency rather than probability can be used in describing risk. Degrees of belief about probability can be chosen as classes or ranks, such as rare/ unlikely/ moderate/ likely/ almost certain, or incredible/ improbable/ remote/ occasional/ probable/ frequent.

3.3026

probability and impact matrix

1. a grid for mapping the probability of each risk occurrence and its impact on project objectives if that risk occurs [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3027

problem

1. difficulty, uncertainty, or otherwise realized and undesirable event, set of events, condition, or situation that requires investigation and corrective action [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.30; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.29] **2.** difficulty or uncertainty experienced by one or more persons, resulting from an unsatisfactory encounter with a system in use [*IEEE 1044-2009 IEEE Standard Classification for Software Anomalies*, 2] **3.** undesirable situation concerning an application, the application management organization, its processes or working methods, which demands structural analysis of the cause and a structural solution [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.27]

Note 1 to entry: A risk factor becomes a problem when a risk metric (an objective measure) crosses a predetermined threshold (the problem trigger). The root cause is not usually known at the time a problem record is created and the problem management process is responsible for further investigation. A problem might concern a service or product or a process (-step) or any other element of the application management organization.

3.3028

problem definition

problem description

1. statement of a problem, which can include a description of the data, the method, the procedures, and algorithms used to solve it [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.3029

problem report (PR)

1. document used to identify and describe problems detected in a product [IEEE 14764-2006 *Software Engineering - Software Life Cycle Processes - Maintenance*, 3.9]

Note 1 to entry: PRs are either submitted directly to denote faults or established after impact analysis is performed on Modification Requests and faults are found.

3.3030

problem state

slave state

user state

1. in the operation of a computer system, a state in which programs other than the supervisory program can execute

cf. supervisor state

3.3031

problem-oriented language

1. programming language designed for the solution of a given class of problems

EXAMPLE: list processing languages, information retrieval languages, simulation languages

3.3032

procedural cohesion

1. type of cohesion in which the tasks performed by a software module all contribute to a given program procedure, such as an iteration or decision process

cf. coincidental cohesion, communicational cohesion, functional cohesion, logical cohesion, sequential cohesion, temporal cohesion

3.3033

procedural language

procedure-oriented language

1. programming language in which the user states a specific set of instructions that the computer must perform in a given sequence

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

Note 1 to entry: All widely-used programming languages are of this type.

3.3034

procedural programming language

1. computer programming language used to express the sequence of operations to be performed by a computer.

cf. nonprocedural programming language

EXAMPLE: COBOL

3.3035

procedure

1. information item that presents an ordered series of steps to perform a process, activity, or task [ISO/IEC/IEEE 15289:2015 *Systems and software engineering — Content of life-cycle information products (documentation)*, 5.19]

2. portion of a computer program that is named and that performs a specific action 3. routine that does not return a value 4. specified way to carry out an activity or process [ISO/IEC 19770-1:2012 *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3. 9] 5. an established method of accomplishing a consistent performance or result. A procedure typically can be described as the

sequence of steps that will be used to execute a process [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: A procedure defines an established and approved way or mode of conducting business in an organization. It details permissible or recommended methods in order to achieve technical or managerial goals or outcomes. When a procedure is specified as an outcome, the resulting deliverable will typically specify what must be done, by whom, and in what sequence. This is a more detailed level of specification than for a process.

3.3036

procedure testing

1. type of functional suitability testing conducted to evaluate whether procedural instructions for interacting with a test item or using its outputs meet user requirements and support the purpose of their use [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.29]

3.3037

process

1. set of interrelated or interacting activities that transforms inputs into outputs [IEEE 730-2014 *IEEE Standard for Software Quality Assurance Processes*; ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.31; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.29] **2.** predetermined course of events defined by its purpose or by its effect, achieved under given conditions [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **3.** to perform operations on data **4.** collection of steps taking place in a prescribed manner [ISO/IEC 15414:2015 *Information technology — Open distributed processing — Reference model — Enterprise language*, 6.3.6] **5.** in data processing, the predetermined course of events that occur during the execution of all or part of a program [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **6.** executable unit managed by an operating system scheduler **7.** system of activities, which use resources to transform inputs into outputs [ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.22] **8.** a systematic series of activities directed towards causing an end result such that one or more inputs will be acted upon to create one or more outputs [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **9.** set of interrelated or interacting activities that use inputs to deliver an intended result [ISO/IEC TR 29110-1:2016 *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.36]

Note 1 to entry: The term "activities" covers use of resources. A process can have multiple starting points and multiple end points. The prescribed manner can be a partially ordered sequence. A process specification can be a workflow specification. An enterprise specification can define types of processes and process templates. A process can be viewed as a specific instantiation of life cycle processes, adapted within a life cycle model, to create the service or product for the specific requirements and context of a project. When a process definition is specified as an outcome, the resulting deliverable typically specifies inputs and outputs, and gives a general description of expected activities. However, it does not include the same level of detail as for a procedure.

3.3038

process action plan

1. plan, usually resulting from appraisals, that documents how specific improvements targeting the weaknesses uncovered by an appraisal will be implemented

3.3039

process action team

1. team that has the responsibility to develop and implement process improvement activities for an organization as documented in a process action plan

3.3040

process analysis

1. A process analysis follows the steps outlined in the process improvement plan to identify needed improvements [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3041

process architect

1. person or group that has primary responsibility for creating and maintaining the software life cycle process (SLCP)

3.3042

process architecture

- 1.** ordering, interfaces, interdependencies, and other relationships among the process elements in a standard process

Note 1 to entry: Process architecture also describes the interfaces, interdependencies, and other relationships between process elements and external processes, such as contract management.

3.3043

process area

- 1.** cluster of related practices in an area that, when implemented collectively, satisfies a set of goals considered important for making improvement in that area

3.3044

process assessment

- 1.** disciplined evaluation of an organizational unit's processes against a process assessment model [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.2.15] **2.** determination of the extent to which the organization's standard processes contribute to the achievement of its business goals and help the organization focus on the need for continuous process improvement

3.3045

process assessment model (PAM)

- 1.** model suitable for the purpose of assessing a specified process quality characteristic, based on one or more process reference models [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.3.9; ISO/IEC TR 29110-1:2016 *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.38]
cf. process reference model (PRM)

Note 1 to entry: Process assessment models addressing a specific process quality characteristic can include the identification of the characteristic in the title; for example, a process assessment model addressing process capability can be termed a "process capability assessment model."

3.3046

process asset library

- 1.** collection of information that can be useful to those who are defining, implementing, and managing processes in the organization.

EXAMPLE: process-related documentation such as policies, defined processes, checklists, lessons-learned documents, templates, standards, procedures, plans, and training materials.

3.3047

process attribute (PA)

process quality attribute

- 1.** measurable property of a process quality characteristic [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.4.3]

3.3048

process attribute outcome

- 1.** observable result of achievement of a specified process attribute [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.4.4]

3.3049

process attribute rating

- 1.** judgment of the degree of achievement of the process attribute for the assessed process [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.4.5]

3.3050

process capability

- 1.** characterization of the ability of a process to meet current or projected business goals [ISO/IEC 33020:2015 *Information technology — Process assessment — Process measurement framework for assessment of process capability*, 3.1] **2.** range of expected results that can be achieved by following a process

3.3051

process capability level

1. characterization of a process on an ordinal measurement scale of process capability [*ISO/IEC 33020:2015 Information technology — Process assessment — Process measurement framework for assessment of process capability, 3.2*] 2. point on the six-point ordinal scale of process capability that represents the capability of the process [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.39*]

Note 1 to entry: Each level builds on the capability of the level below.

3.3052

process component

1. CORBA component with persistent state, which is not visible to the client, persistent identity, and behavior, which can be transactional [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

3.3053

process decision program chart (PDPC)

1. The PDPC is used to understand a goal in relation to the steps for getting to the goal [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3054

process definition

1. identification of a sequence of steps involving activities, constraints, and resources that are performed for a given purpose

3.3055

process description

1. documented expression of a set of activities performed to achieve a given purpose

Note 1 to entry: A process description provides an operational definition of the major components of a process. The description specifies, in a complete, precise, and verifiable manner, the purpose, outcomes, activities and tasks, requirements, design, behavior, or other characteristics of a process. It also can refer to procedures for determining whether these provisions have been satisfied. Process descriptions can be applicable at the project or organizational level.

3.3056

process dimension

1. set of process elements in a process assessment model explicitly related to the processes defined in the relevant process reference model(s) [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.3.10*]

Note 1 to entry: The elements of the process dimension include processes, process purpose statements, process outcomes, and process performance indicators.

3.3057

process group

1. collection of related processes 2. team of specialists who facilitate the definition, maintenance, and improvement of processes used by the organization

3.3058

process improvement

1. actions taken to improve the quality of the organization's processes aligned with the business needs and the needs of other concerned parties [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.1.7*] 2. result of activities that better the performance and maturity of the organization's processes 3. actions taken to improve the quality of the organization's processes aligned with the business needs [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 3.41*]

3.3059

process improvement objective

1. set of target characteristics established to guide the effort to improve an existing process in a specific, measurable way, either in terms of resultant product or service characteristics, such as quality, performance, and

conformance to standards, or in the way in which the process is executed, such as elimination of redundant process steps, combination of process steps, and improvement of cycle time

3.3060

process improvement plan

1. a subsidiary plan of the project management plan. It details the steps for analyzing processes to identify activities that enhance their value [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3061

process improvement support element

1. way that an organization expresses support for process improvement projects or initiatives [*ISO/IEC TR 33014:2013 Information technology — Process assessment — Guide for process improvement, 3.2*]

3.3062

process infrastructure

1. internal structure of the software life-cycle process, to include lifecycle phases, documentation, baselines, reviews, and products

3.3063

process instance

1. single specific and identifiable execution of a process [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.7; ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.2.17*]

3.3064

process management

1. direction, control, and coordination of work performed to develop a product or perform a service

EXAMPLE: quality assurance

3.3065

process measurement framework

1. schema for use in characterizing a process quality characteristic of an implemented process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology, 3.4.6*]

3.3066

process metric

1. metric used to measure characteristics of the methods, techniques, and tools employed in developing, implementing, and maintaining the software system [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.15*]

3.3067

process outcome

1. observable result of the successful achievement of the process purpose [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.27; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.32*]

Note 1 to entry: An outcome is an artifact, a significant change of state or the meeting of specified constraints. An outcome statement describes one of the following: production of an artifact; a significant change in state; meeting of specified constraints, e.g., requirements, goals.

3.3068

process owner

1. person (or team) responsible for defining and maintaining a process

Note 1 to entry: At the organizational level, the process owner is the person (or team) responsible for the description of a standard process; at the project level, the process owner is the person (or team) responsible for the description of the defined process. A process can therefore have multiple owners at different levels of responsibility.

3.3069

process performance

- 1.** extent to which the execution of a process achieves its purpose [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.7]

3.3070

process performance indicator

- 1.** assessment indicator that supports the judgment of the process performance of a specific process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.3.12]

3.3071

process profile

- 1.** set of process attribute ratings for an assessed process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.2.18]

3.3072

process purpose

- 1.** high-level objective of performing the process and the likely outcomes of effective implementation of the process [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.33; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.31]

Note 1 to entry: The implementation of the process is intended to provide tangible benefits to the stakeholders.

3.3073

process quality

- 1.** ability of a process to satisfy stated and implied stakeholder needs when used in a specified context [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.8]

3.3074

process quality characteristic

- 1.** measurable aspect of process quality [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.9] **2.** category of process attributes that are significant to process quality [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.9]

3.3075

process quality determination

- 1.** systematic assessment and analysis of selected processes against a target process profile [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.2.19]

3.3076

process quality dimension

- 1.** set of elements in a process assessment model explicitly related to the process measurement framework for the specified process quality characteristic [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.3.14]

3.3077

process quality indicator

- 1.** assessment indicator that supports the judgment of the process quality characteristic of a specific process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.3.15]

3.3078

process quality level

- 1.** point on a scale of achievement of a process quality characteristic derived from the process attribute ratings for an assessed process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.4.10] **2.** representation of the achieved level of a process quality characteristic derived from the process attribute ratings for an assessed process [*ISO/IEC TR 29110-3-1:2015 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-1: Assessment Guide*, 3.2]

3.3079

process reference model (PRM)

1. model comprising definitions of processes in a domain of application described in terms of process purpose and outcomes, together with an architecture describing the relationships between the processes [ISO/IEC 33001:2015 *Information technology — Process assessment — Concepts and terminology*, 3.3.16] 2. model comprising definitions of processes in a lifecycle described in terms of process purpose and outcomes, together with an architecture describing the relationships between the processes [ISO/IEC TR 29110-1:2016 *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.44]
cf. process assessment model (PAM)

3.3080

process standard

1. standard that deals with the series of actions or operations used in making or achieving a product

3.3081

process tailoring

1. making, altering, or adapting a process description for a particular end

EXAMPLE: A project tailors its defined process from the organization's set of standard processes to meet objectives, constraints, and the environment of the project.

Note 1 to entry: For example,

3.3082

process view

1. description of how a specified purpose and set of outcomes can be achieved by employing the activities and tasks of existing processes [ISO/IEC 15026-1:2013 *Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*, 3.2.2]

3.3083

processing logic

1. any of the requirements specifically requested by the user to complete an elementary process, such as validations, algorithms, or calculations, and reading or maintaining a file [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*]

3.3084

processor

1. in a computer, a functional unit that interprets and executes instructions [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

Note 1 to entry: A processor consists of at least an instruction control unit and an arithmetic and logic unit

3.3085

procurement audit

1. the review of contracts and contracting processes for completeness, accuracy and effectiveness [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3086

procurement documents

1. the documents utilized in bid and proposal activities, which include the buyer's Invitation for Bid, Invitation for Negotiations, Request for Information, Request for Quotation, Request for Proposal and seller's responses [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3087

procurement management plan

1. a component of the project or program management plan that describes how a project team will acquire goods and services from outside the performing organization [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3088

procurement performance review

1. a structured review of the seller's progress to deliver project scope and quality, within cost and on schedule, as compared to the contract [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3089

procurement statement of work

1. describes the procurement item in sufficient detail to allow prospective sellers to determine if they are capable of providing the products, services, or results [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3090

producer object

1. object which is the source of the information conveyed [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.4.3*]

3.3091

product

1. an artifact that is produced, is quantifiable, and can be either an end item in itself or a component item [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** complete set of computer programs, procedures, and associated documentation designed for delivery to a user [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation, 3.30*] **3.** result of a process [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.32; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.34; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.32*] **4.** output of the software development activities (e.g., document, code, or model) **5.** part of the equipment (hardware, software and materials) for which usability is to be specified or evaluated [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuARE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.9*] *cf.* activity, deliverable, result

Note 1 to entry: Generic product categories are hardware (e.g., engine mechanical part); software (e.g., computer program); services (e.g., transport); and processed materials (e.g., lubricant). Hardware and processed materials are generally tangible products, while software or services are generally intangible. Most products comprise elements belonging to different generic product categories. Whether the product is then called hardware, processed material, software, or service depends on the dominant element. Results could be components, systems, software, services, rules, documents, or many other items. The result could in some cases be many related individual results.

3.3092

product analysis

1. process of evaluating a product by manual or automated means to determine if the product has certain characteristics **2.** For projects that have a product as a deliverable, it is a tool to define scope that generally means asking questions about a product and forming answers to describe the use, characteristics and other the relevant aspects of what is going to be manufactured [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3093

product authority

1. person or persons with overall responsibility for the capabilities and quality of a product [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.37*]

3.3094

product baseline

product configuration baseline

production configuration

1. description of the detailed design at a specific point in time, for production, fielding/deployment, and operations and support [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1*] *cf.* allocated baseline, developmental configuration, functional baseline, product configuration identification

EXAMPLE: The product baseline prescribes all necessary physical (form, fit, or function) characteristics and selected functional characteristics designated for production acceptance testing and production test requirements.

3.3095

product configuration identification

- 1.** current approved or conditionally approved technical documentation defining a configuration item during the production, operation, maintenance, and logistic support phases of its life cycle
cf. allocated configuration identification, functional configuration identification, product baseline

Note 1 to entry: It prescribes all necessary physical or form, fit, and function characteristics of a configuration item, the selected functional characteristics designated for production acceptance testing, and the production acceptance tests.

3.3096

product description

- 1.** document stating properties of software, with the main purpose of helping potential acquirers in the evaluation of the suitability for themselves of the software before purchasing it [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.44*]

Note 1 to entry: The product description is not a specification; it serves a different purpose.

3.3097

product engineering

- 1.** technical processes to define, design, and construct or assemble a product

3.3098

product identification

- 1.** software product name, version, variant, and date information [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.12*]

3.3099

product life cycle

- 1.** the series of phases that represent the evolution of a product, from concept through delivery, growth, maturity, to retirement [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3100

product line

product family

- 1.** set of products or services sharing explicitly defined and managed common and variable features and relying on the same domain architecture to meet the common and variable needs of specific markets [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.16*]

3.3101

product line platform

- 1.** product line architecture, a configuration management plan, and domain assets, enabling application engineering to effectively reuse and produce a set of derivative products [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.18*]

3.3102

product line reference model

- 1.** abstract representation of the domain and application engineering life cycle processes; the roles and relationships of the processes; and the assets produced, managed, and used during product line engineering and management [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.19*]

3.3103

product line scoping

- 1.** process for defining the member products that will be produced within a product line and the major common and variable features among the products, analyzes the products from an economic point of view, and controls

and schedules the development, production, and marketing of the product line and its products [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.20]

3.3104

product management

1. definition, coordination, and control of the characteristics of a product during its development cycle

EXAMPLE: configuration management

3.3105

product metric

1. metric used to measure the characteristics of any intermediate or final product of the software development process [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.16]

3.3106

product quality

1. degree to which the product satisfies stated and implied needs when used under specified conditions [*ISO/IEC 25041: 2012 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation guide for developers, acquirers and independent evaluators*, 4.11]

Note 1 to entry: This definition differs from the ISO 9000:2015 quality definition, because this definition refers to the satisfaction of stated and implied needs, while the ISO 9000 quality definition refers to the satisfaction of requirements.

3.3107

product requirement

1. refinement of customer requirements into the developers' language, making implicit requirements into explicit derived requirements

3.3108

product risk

1. risk that a product could be defective in some specific aspect of its function, quality, or structure [*ISO/IEC/IEEE 29119-3:2013 Software and systems engineering — Software testing — Part 3: Test documentation*, 4.8]

3.3109

product scope

1. the features and functions that characterize a product, service or result [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3110

product scope description

1. the documented narrative description of the product scope [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3111

product scoping

1. subprocess of product line scoping that determines the product roadmap, that is 1) the targeted markets; 2) the product categories that the product line organization should be developing, producing, marketing, and selling; 3) the common and variable features that the products should provide in order to reach the long and short-term business objectives of the product line organization, and 4) the schedule for introducing products to markets [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.20]

3.3112

product specification

1. document that specifies the design that production copies of a system or component must implement **2.** document that describes the characteristics of a planned or existing product for consideration by potential customers or users

cf. design description

Note 1 to entry: For software, this document describes the as-built version of the software.

3.3113

product standard

- 1.** standard that defines what constitutes completeness and acceptability of items that are used or produced, formally or informally, during the software engineering process

3.3114

product support

- 1.** providing of information, assistance, and training to install and make software operational in its intended environment and to distribute improved capabilities to users

3.3115

production

- 1.** stage in the life cycle when completed software or documentation is prepared, published, and packaged for distribution

3.3116

production library

- 1.** software library containing software approved for current operational use
cf. master library, software development library, software repository, system library

3.3117

production plan

- 1.** description of how domain assets are to be used to develop member products in a product line [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.17*]

3.3118

production rate

- 1.** a measure of the amount of work completed per unit of time, such as user stories or features per week
[*Software Extension to the PMBOK® Guide Fifth Edition*]
cf. burndown rate, velocity

3.3119

productivity

- 1.** ratio of work product to work effort

3.3120

professional standard

- 1.** standard that identifies a profession as a discipline and distinguishes it from other professions

3.3121

profile

- 1.** set of one or more base standards or profiles and, where applicable, the identification of chosen classes, conforming subsets, options and parameters of those base standards or standardized profiles necessary to accomplish a particular function [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 2.23*]

Note 1 to entry: [ISO/IEC TR 10000-1]

3.3122

profile group

PG

- 1.** collection of profiles which are related by composition of processes (i.e., activities, tasks) or by capability level, or both [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 2.24*] **2.** collection of profiles which are related by composition of processes (i.e., activities, tasks), or by requirements sharing or composition [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.45*]

3.3123

program

- 1.** to write a computer program **2.** to design, write, modify, and test programs [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **3.** a group of related projects, subprograms and program activities managed in a coordinated way to obtain benefits not available from managing them individually [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. computer program

3.3124

program construct

- 1.** set of one or more procedure parts and a control part which can be implicit [*ISO/IEC 8631:1989 Information technology — Program constructs and conventions for their representation, 2*]

Note 1 to entry: Each procedure part consists of one or more operations to be performed or can be null.

3.3125

program design language (PDL)

- 1.** specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a program design
cf. hardware design language, pseudo code

3.3126

Program Evaluation and Review Technique (PERT)

- 1.** a technique for estimating that applies a weighted average of optimistic, pessimistic, and most likely estimates when there is uncertainty with the individual activity estimates [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3127

program instruction

- 1.** computer instruction in a source program

Note 1 to entry: A program instruction is distinguished from a computer instruction that results from assembly, compilation, or other interpretation process.

3.3128

program librarian

- 1.** person responsible for establishing, controlling, and maintaining a software development library

3.3129

program listing

- 1.** printout or other human readable display of the source and, sometimes, object statements that make up a computer program

3.3130

program maintenance manual

- 1.** document that provides the information necessary to maintain a program [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3131

program management

- 1.** the application of knowledge, skills, tools, and techniques to a program to meet the program requirements and to obtain benefits and control not available by managing projects individually. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3132

program mutation

- 1.** computer program that has been purposely altered from the intended version to evaluate the ability of test cases to detect the alteration **2.** process of creating an altered program as in (1)
cf. mutation testing

3.3133

program network chart

1. diagram that shows the relationship between two or more computer programs

3.3134

program specification

1. document that describes the structure and functions of a program in sufficient detail to permit programming and to facilitate maintenance [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.3135

program status word (PSW)

1. computer word that contains information specifying the current status of a computer program 2. special-purpose register that contains a program status word as in (1)

Note 1 to entry: The information can include error indicators, the address of the next instruction to be executed, or currently enabled interrupts.

3.3136

program synthesis

1. use of software tools to aid in the transformation of a program specification into a program that realizes that specification

3.3137

program-sensitive fault

1. fault that causes a failure when some particular sequence of program steps is executed

cf. data-sensitive fault

3.3138

programmable breakpoint

1. breakpoint that automatically invokes a previously specified debugging process when initiated

cf. code breakpoint, data breakpoint, dynamic breakpoint, epilog breakpoint, prolog breakpoint, static breakpoint

3.3139

programmable counter array

1. group of counter modules on a microcontroller unit that increases its timing capabilities

EXAMPLE: The parameters for counting are configured programmatically.

3.3140

programmable pulse generator

1. part of a microcontroller unit that produces clock signals, adjustable through programming

3.3141

programmable read-only memory (PROM)

1. read-only memory unit without a rewrite or an erase data function, which the user can program only once

3.3142

programmable reload timer (PRT)

1. reconfigurable device to restart a function, using decrementing, status control, and reload registers

3.3143

programmable terminal

intelligent terminal

1. user terminal that has built-in data processing capability [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.3144

programmatic reference point

1. reference point at which a programmatic interface can be established to allow access to a function [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 15.3.1]

3.3145

programmer manual

- 1.** document that provides the information necessary to develop or modify software for a given computer system
cf. diagnostic manual, installation manual, operator manual, support manual, user manual

Note 1 to entry: Typically described are the equipment configuration, operational characteristics, programming features, input/output features, and compilation or assembly features of the computer system.

3.3146

programming

- 1.** general activity of software development **2.** designing, writing, modifying, and testing of programs [*ISO/IEC 2382:2015, Information technology — Vocabulary*]
cf. construction

Note 1 to entry: especially construction activities.

3.3147

programming language

- 1.** language used to express computer programs **2.** artificial language for expressing programs [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3148

programming support environment

- 1.** integrated collection of software tools accessed via a single command language to provide programming support capabilities throughout the software life cycle
cf. scaffolding

Note 1 to entry: sometimes called integrated programming support environment. The environment typically includes tools for specifying, designing, editing, compiling, loading, testing, configuration management, and project management.

3.3149

programming system

- 1.** set of programming languages and the support software (editors, compilers, linkers, etc.) necessary for using these languages with a given computer system

3.3150

progressive elaboration

- 1.** the iterative process of increasing the level of detail in a project management plan as greater amounts of information and more accurate estimates become available [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3151

prohibition

- 1.** prescription that a particular behavior must not occur [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.6*]

3.3152

project

- 1.** endeavor with defined start and finish criteria undertaken to create a product or service in accordance with specified resources and requirements [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.33; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.35; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.33*] **2.** undertaking with pre-specified objectives, magnitude and duration [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **3.** a temporary endeavor undertaken to create a unique product, service, or result [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **4.** set of activities for developing a new product or enhancing an existing product [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.38*]

Note 1 to entry: A project is sometimes viewed as a unique process comprising coordinated and controlled activities and can be composed of activities from the Project Processes and Technical Processes.

3.3153

project balance

- 1.** representation, as a series of cash amounts at regular intervals, of the cumulative to-date value of an alternative

3.3154

project calendar

- 1.** a calendar that identifies working days and shifts that are available for scheduled activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. resource calendar

3.3155

project charter

- 1.** a document issued by the project initiator or sponsor that formally authorizes the existence of a project, and provides the project manager with the authority to apply organizational resources to project activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3156

project communications management

- 1.** Project communications management includes the processes that are required to ensure timely and appropriate planning, collection, creation, distribution, storage, retrieval, management, control, monitoring, and the ultimate disposition of project information [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. information management

3.3157

project control

- 1.** activities concerned with monitoring the progress of a project, its direction, quality, and resource utilization, as compared with project plan [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3158

project cost management

- 1.** Project cost management includes the processes involved in planning, estimating, budgeting, financing, funding, managing, and controlling costs so that the project can be completed within the approved budget [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3159

project file

project notebook

- 1.** central repository of material pertinent to a project

Note 1 to entry: Contents typically include memos, plans, technical reports, and related items.

3.3160

project function point count

- 1.** the size of a development project or an enhancement project 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*]

Note 1 to entry: In other words, the total functionality to be added, changed, or deleted. It enables those involved to determine the effort required in order to realize new software or to change the functionality of existing software. In the latter case, a project function point count pertains to the addition, change, or deletion of functions.

3.3161

project funding requirements

- 1.** forecast project costs to be paid for that are derived from the cost baseline for total or periodic requirements, including projected expenditures plus anticipated liabilities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3162

project governance

- 1.** the alignment of project objectives with the strategy of the larger organization by the project sponsor and project team. A project's governance is defined by and must fit within the larger context of the program or

350

organization sponsoring it, but is separate from organizational governance [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3163

project human resource management

1. Project Human Resource Management includes the processes that organize, manage, and lead the project team [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3164

project initiation

1. launching a process that can result in the authorization of a new project [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3165

project integration management

1. Project integration management includes the processes and activities needed to identify, define, combine, unify, and coordinate the various processes and project management activities within the project management process groups [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3166

project life cycle

1. the series of phases that a project passes through from its initiation to its closure [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3167

project management (PM)

1. the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** activities concerned with project planning and project control [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.3168

Project Management Body of Knowledge (PMBOK®)

1. An inclusive term that describes the sum of knowledge within the profession of project management. As with other professions, such as law, medicine, and accounting, the body of knowledge rests with the practitioners and academics that apply and advance it. The complete project management body of knowledge includes proven traditional practices that are widely applied and innovative practices that are emerging in the profession. The body of knowledge includes both published and unpublished materials. This body of knowledge is constantly evolving [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3169

project management information system (PMIS)

1. an information system consisting of the tools and techniques used to gather, integrate, and disseminate the outputs of project management processes. It is used to support all aspects of the project from initiating through closing, and can include both manual and automated systems [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3170

project management knowledge area

1. an identified area of project management defined by its knowledge requirements and described in terms of its component processes, practices, inputs, outputs, tools, and techniques [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3171

project management office (PMO)

1. an organizational structure that standardizes the project-related governance processes and facilitates the sharing of resources, methodologies, tools and techniques [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3172

project management plan

1. the document that describes how the project will be executed, monitored, and controlled [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3173

project management process group

1. a logical grouping of project management inputs, tools and techniques, and outputs. The project management process groups include initiating processes, planning processes, executing processes, monitoring and controlling processes, and closing processes. Project management process groups are not project phases [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3174

project management software

1. applications specifically designed to aid the project management team with planning, monitoring, and controlling the project, including cost estimating, scheduling, collaboration, and risk analysis

3.3175

project management staff

1. the members of the project team who perform project management activities such as schedule, communications, risk management, etc. [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3176

project management system

1. the aggregation of the processes, tools, techniques, methodologies, resources, and procedures to manage a project [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3177

project management team

1. the members of the project team who are directly involved in project management activities. On some smaller projects, the project management team may include virtually all of the project team members [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3178

project manager (PM)

PJM

1. the person assigned by the performing organization to lead the team that is responsible for achieving the project objectives [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 2. person with overall responsibility for the management and running of a project [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.39]

3.3179

project organization chart

1. a document that graphically depicts the project team members and their interrelationships for a specific project [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3180

project performance

1. derived measure that gives an indication of some attribute associated with how well, how quickly, how effectively, or how efficiently a project is carried out [ISO/IEC 29155-1:2011 *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.8]

3.3181

project phase

1. a collection of logically related project activities that culminates in the completion of one or more deliverables [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. stage

3.3182

project plan

- 1.** document that describes the technical and management approach to be followed for a project

Note 1 to entry: for example, a software development plan. The plan typically describes the work to be done, the resources required, the methods to be used, the procedures to be followed, the schedules to be met, and the way that the project will be organized.

3.3183

project planning

- 1.** activities concerned with the specification of the components, timing, resources, and procedures of a project [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3184

project portfolio

- 1.** collection of projects that addresses the strategic objectives of the organization [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.30*]

3.3185

project procurement management

- 1.** Project procurement management includes the processes necessary to purchase or acquire products, services, or results needed from outside the project team [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3186

project quality management

- 1.** Project quality management includes the processes and activities of the performing organization that determine quality policies, objectives, and responsibilities so that the project will satisfy the needs for which it was undertaken [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3187

project resource constraint

- 1.** limitation or restraint placed on resource usage, such as what resource skills or disciplines are available and the amount of a given resource available during a specified time frame

3.3188

project risk

- 1.** risk related to the management of a project [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.31*]

EXAMPLE: lack of staffing, strict deadlines, changing requirements

3.3189

project risk management

- 1.** project risk management includes the processes of conducting risk management planning, identification, analysis, response planning, and controlling risk on a project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3190

project risk profile

- 1.** project's current and historical risk-related information; a compendium or aggregate of all of the individual risk profiles in a project [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.4*]

cf. risk profile, risk state

Note 1 to entry: The project risk profile information includes the risk management context, along with the chronological record of risks and their individual risk profiles, priority ordering, risk-related measures, treatment status, contingency plans, and risk action requests. A project risk profile consists of a collection of the risk profiles of all the individual risks, which in turn includes the current and historical risk states.

3.3191

project schedule

1. an output of a schedule model that presents linked activities with planned dates, durations, milestones, and resources [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3192

project schedule network diagram

activity network diagram

1. a graphical display of the logical relationships among the project schedule activities [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3193

project scope

1. the work that must be performed to deliver a product, service, or result with the specified features and functions [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3194

project scope management

1. Project scope management includes the processes required to ensure that the project includes all the work required, and only the work required, to complete the project successfully [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3195

project scope statement

1. the description of the project scope, major deliverables, assumptions, and constraints [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3196

project specification

1. specification of the objectives, requirements, and scope of a project and its relations to other projects [ISO/IEC 2382:2015, *Information technology — Vocabulary* 20.07.07]

3.3197

project stakeholder management

1. Project stakeholder management includes the processes required to identify all people or organizations impacted by the project, analyzing stakeholder expectations and impact on the project, and developing appropriate management strategies for effectively engaging stakeholders in project decisions and execution [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3198

project team

1. project manager, and, for some projects, the project sponsor, and the group of persons who report either directly or indirectly to the project manager and who are responsible for performing project work as a regular part of their assigned duties, including the staff supporting project management 2. a set of individuals who support the project manager in performing the work of the project to achieve its objectives [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3199

project team directory

1. a documented list of project team members, their project roles and communication information [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3200

project time management

1. Project time management includes the processes required to manage the timely completion of a project [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3201

project-based organizations

1. a variety of organizational forms that involve the creation of temporary systems for the performance of projects. PBOs conduct the majority of their activities as projects and/or provide project over functional approaches [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3202

projectized organization

1. any organizational structure in which the project manager has full authority to assign priorities, apply resources, and direct the work of persons assigned to the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3203

prolog breakpoint

preamble breakpoint

1. breakpoint that is initiated upon entry into a program or routine

cf. epilog breakpoint, code breakpoint, data breakpoint, dynamic breakpoint, programmable breakpoint, static breakpoint

3.3204

PROM

1. programmable read-only memory

3.3205

prompt

1. symbol or message displayed by a computer system, requesting input from the user of the system **2.** to display a symbol or message as in (1) **3.** visual or audible message sent by a program to request or guide the user's response [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3206

proof of correctness

1. formal technique used to prove mathematically that a computer program satisfies its specified requirements **2.** proof that results from applying the technique in (1)

cf. assertion, formal specification, inductive assertion method, partial correctness, total correctness

3.3207

property

1. responsibility that is an inherent or distinctive characteristic or trait that manifests some aspect of an object's knowledge or behavior [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.151*] **2.** measurable phenomenological characteristic of an interaction [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.22*]

Note 1 to entry: Three kinds of property are defined: attribute, participant properties due to relationships, and operations.

3.3208

property state

1. state comprising properties associated with a unit that are defined by functional dependence on the properties of past interaction occurrences and that affect the properties of future output interaction occurrences [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.23*]

Note 1 to entry: A particular property state of the unit is defined at a point in time by the values of the internal state properties.

3.3209

property to quantify

1. property of a target entity that is related to a quality measure element and which can be quantified by a measurement method [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.11*]

3.3210

property-of-interest

- 1.** any property that, if lost, is considered a negative effect [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 1*]

Note 1 to entry: The concept of property-of-interest is introduced in order to characterize negative effects of consequences. In the safety context, human lives and health are instances of properties-of-interest. Assets in the security context are instances of properties-of-interest.

3.3211

proposal

- 1.** supplier's offer to provide a system or service, usually including benefits, costs, risks, opportunities, and other factors applicable to decisions

Note 1 to entry: includes business cases

3.3212

proposal evaluation techniques

- 1.** the process of reviewing proposals provided by suppliers to support contract award decisions [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3213

proposed change

- 1.** report of anomaly, required or recommended enhancement from the time an idea is recorded until the disposition by a designated change authority [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management, 4.3*]

Note 1 to entry: The disposition can be to reject, to defer for further analysis, or to accept. Upon acceptance the proposed change becomes an approved modification. There can be a one-to-one, one-to-many, or many-to-many relationship between proposed changes and approved modifications.

3.3214

proposition

- 1.** observable fact or state of affairs involving one or more entities, of which it is possible to assert or deny that it holds for those entities [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 6.2*]

3.3215

protected

- 1.** responsibility that is visible only to the class or the receiving instance of the class (available only within methods of the class or its subclasses) [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.152*]

cf. private, public, hidden

3.3216

protection exception

- 1.** exception that occurs when a program attempts to write into a protected area in storage
cf. addressing exception, data exception, operation exception, overflow exception, underflow exception

3.3217

protocol

- 1.** set of conventions that govern the interaction of processes, devices, and other components within a system

3.3218

protocol object

- 1.** engineering object in a channel, which communicates with other protocol objects in the same channel to achieve interaction between basic engineering objects (possibly in different clusters, possibly in different capsules, possibly in different nodes) [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.12*]

3.3219

prototype

- 1.** preliminary type, form, or instance of a system that serves as a model for later stages or for the final, complete version of the system
2. a method of obtaining early feedback on requirements by providing a working model of the expected product before actually building it [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: A prototype is used to get feedback from users for improving and specifying a complex human interface, for feasibility studies, or for identifying requirements.

3.3220

prototyping

- 1.** hardware and software development technique in which a preliminary version of part or all of the hardware or software is developed to permit user feedback, determine feasibility, or investigate timing or other issues in support of the development process
cf. rapid prototyping

3.3221

provision

- 1.** expression in the content of a normative document, that takes the form of a statement, an instruction, a recommendation or a requirement [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1, 3.8*]

Note 1 to entry: These types of provision are distinguished [in English] by the form of wording they employ, e.g., instructions are expressed in the imperative mood, recommendations by the use of the auxiliary "should", and requirements by the use of the auxiliary "shall".

[SOURCE: ISO/IEC Guide 2:2004]

3.3222

proxy home

- 1.** implementation of the component home interface specified by a composition definition [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

Note 1 to entry: The implementation is not required to be collocated with the container where the components managed by the home are activated.

3.3223

PRR

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

3.3224

PRT

- 1.** programmable reload timer

3.3225

PSCI

- 1.** Protocol Support for Computational Interactions [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 4*]

3.3226

PSDL

- 1.** persistent state definition language [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.3227

pseudo instruction

pragma

pseudo-op

pseudo operation

1. source language instruction that provides information or direction to the assembler or compiler and is not translated into a target language instruction

EXAMPLE: an instruction specifying the desired format of source code listings

3.3228

pseudocode

1. combination of programming language constructs and natural language used to express a computer program design **2.** general term for structured English or program design language **3.** English-like statements used for low-level program design

EXAMPLE: IF the data arrives faster than expected, THEN reject every third input ELSE process all data received ENDIF.

3.3229

pseudostatic random access memory (PSRAM)

1. dynamic random access memory unit with an automatic refresh circuit on the unit

3.3230

PSM

1. platform-specific model [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.3231

PSP

1. product support plan [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.3232

PSRAM

1. pseudostatic random access memory

3.3233

PSW

1. program status word

3.3234

PTNG

1. Place/Transition Net Graph [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.2.6]

3.3235

public

1. responsibility that is not hidden [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.153] **2.** known to multiple routines or modules cf. hidden, private, protected

Note 1 to entry: That is, visible to any requester (available to all without restriction).

3.3236

publisher

1. event source that can be connected to an arbitrary number of event sinks, which subscribe to the publisher event source [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

3.3237

pulse width modulation (PWM)

1. control of electrical signals with various cycle and duty ratios

3.3238

purpose of the count

1. reason for performing the function point count [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.45*]

3.3239

purpose statement

1. brief statement of the reason for an IDEF0 model's existence that is presented in the a-0 context diagram of the model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.96*]

3.3240

PV

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

3.3241

PWM

1. pulse width modulation

3.3242

QA

1. quality assurance

3.3243

QC

1. quality control

3.3244

QCD

1. quality, cost, delivery [*ISO/IEC TR 29110-2-2:2011 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-2: Guide for the development of domain-specific profiles, 4*]

3.3245

QE

1. quality engineering [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.3*]

3.3246

QFD

1. quality function deployment [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3247

QFP

1. quad flat pack

3.3248

QM

1. quality management **2.** quality measure [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 5 b)*]

3.3249

QME

1. quality measure element [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 5*]

3.3250

QMS

1. quality management system

Note 1 to entry: [ISO 9000:2005]

3.3251

QOS

1. Quality of Service [*ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications*, 4]

3.3252

QR

1. quality requirements [*ISO/IEC TR 14143-4:2002 Information technology — Software measurement — Functional size measurement — Part 4: Reference model*, 4]

3.3253

quad flat pack (QFP)

1. surface mount rectangular circuit package with gull-wing shaped leads extending from each of the four sides

Note 1 to entry: A low profile QFP (LQFP) is thinner than a QFP.

3.3254

qualification

1. process of demonstrating whether an entity is capable of fulfilling specified requirements [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.31; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.36] **2.** process of determining whether a system or component is suitable for operational use

3.3255

qualification body

1. entity issuing certificates of qualification in conformance with ISO/IEC 24773:2008 [*ISO/IEC 24773:2008 Software engineering -Certification of software engineering professionals -Comparison framework*, 3.4]

3.3256

qualification requirement

1. set of criteria or conditions that have to be met in order to qualify a software product as complying with its specifications and being ready for use in its target environment or integration with its containing system [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.32]

3.3257

qualification testing

1. testing, conducted by the developer and witnessed by the acquirer (as appropriate), to demonstrate that a software product meets its specifications and is ready for use in its target environment or integration with its containing system [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.33] **2.** testing conducted to determine whether a system or component is suitable for operational use.
cf. acceptance testing, development testing, operational testing

3.3258

qualitative risk analysis

1. prioritizing risks for subsequent further analysis or action by assessing and combining their probability of occurrence and impact

3.3259

quality

1. degree to which the system satisfies the stated and implied needs of its various stakeholders, and thus provides value [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 3.1] **2.** ability of a product, service, system, component, or process to meet customer or user needs, expectations, or requirements **3.** the degree to which a set of inherent characteristics fulfils requirements [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3260

quality assurance (QA)

1. part of quality management focused on providing confidence that quality requirements will be fulfilled [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.37; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*,

4.1.34] **2.** all the planned and systematic activities implemented within the quality system, and demonstrated as needed, to provide adequate confidence that an entity will fulfill requirements for quality [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.34]

Note 1 to entry: [ISO 9000:2005] There are both internal and external purposes for quality assurance: within an organization, quality assurance provides confidence to management; in contractual situations, quality assurance provides confidence to the customer or others. Some quality control and quality assurance actions are interrelated. Unless requirements for quality fully reflect the needs of the user, quality assurance does not necessarily provide adequate confidence.

3.3261

quality attribute

1. characteristic of software, or a generic term applying to quality factors, quality subfactors, or metric values [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.17] **2.** feature or characteristic that affects an item's quality **3.** requirement that specifies the degree of an attribute that affects the quality that the system or software must possess

3.3262

quality audit

1. a quality audit is a structured, independent process to determine if project activities comply with organizational and project policies, processes, and procedures [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3263

quality characteristic

1. category of product attributes that bears on quality **2.** inherent characteristic of a product, process, or system related to a requirement [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.38; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.35]

Note 1 to entry: Critical quality characteristics commonly include those related to health, safety, security, assurance, reliability, availability and supportability.

3.3264

quality checklist

1. a structured tool used to verify that a set of required steps has been performed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3265

quality control (QC)

1. set of activities designed to evaluate the quality of developed or manufactured products **2.** monitoring service performance or product quality, recording results, and recommending necessary changes
cf. quality assurance

Note 1 to entry: This term has no standardized meaning in software engineering at this time.

3.3266

quality control measurements

1. the documented results of control quality activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3267

quality evaluation

1. systematic examination of the extent to which an entity is capable of fulfilling specified requirements

Note 1 to entry: The requirements can be formally specified, as when a product is developed for a specific user under a contract, or specified by the development organization, as when a product is developed for unspecified users, such as consumer software, or the requirements may be more general, as when a user evaluates products for comparison and selection purposes.

3.3268

quality factor

1. management-oriented attribute of software that contributes to its quality [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.18] 2. higher-level quality attribute

3.3269

quality factor sample

1. set of quality factor values that is drawn from the metrics database and used in metrics validation [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.19]

3.3270

quality factor value

1. value of the direct metric that represents a quality factor [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.2]

cf. metric value

3.3271

quality function deployment (QFD)

1. a facilitated workshop technique that helps determine critical characteristics for new product development [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3272

quality in use (measure)

1. extent to which a product used by specific users meets their needs to achieve specific goals with effectiveness, productivity, safety and satisfaction in specific contexts of use [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.42] 2. degree to which a product or system can be used by specific users to meet their needs to achieve specific goals with effectiveness, efficiency, freedom from risk and satisfaction in specific contexts of use [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.24; *ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.8]

cf. usability

Note 1 to entry: This definition of quality in use is similar to the definition of usability in ISO 9241-11. Before the product is released, quality in use can be specified and measured in a test environment designed and used exclusively by the intended users for their goals and contexts of use, e.g. User Acceptance Testing Environment.

3.3273

quality in use measure

1. measure of the degree to which a product or system can be used by specific users to meet their needs to achieve specific goals with effectiveness, efficiency, freedom from risk, satisfaction and content coverage in specific contexts of use [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.25; *ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*, 4.12]

3.3274

quality management

1. coordinated activities to direct and control an organization with regard to quality [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.39; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.36]

Note 1 to entry: ISO 9000-2000

3.3275

quality management and control tools

1. They are a type of quality planning tools used to link and sequence the activities identified [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3276

quality management plan

1. a component of the project or program management plan that describes how an organization's quality policies will be implemented [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3277

quality management system

1. The organizational framework whose structure provides the policies, processes, procedures, and resources required to implement the quality management plan. The typical project quality management plan should be compatible to the organization's quality management system [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3278

quality measure

QM

1. measure that is defined as a measurement function of two or more values of quality measure elements [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.3.10*] **2.** derived measure that is defined as a measurement function of two or more values of quality measure elements [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.13*] cf. software quality measure

3.3279

quality measure element (QME)

1. measure defined in terms of a property and the measurement method for quantifying it, including optionally the transformation by a mathematical function [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.26; ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.14*] **2.** measure defined in terms of an attribute and the measurement method for quantifying it, including optionally the transformation by a mathematical function [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.3.11*]

Note 1 to entry: The software quality characteristics or subcharacteristics of the entity are derived afterwards by calculating a software quality measure.

3.3280

quality metric

1. quantitative measure of the degree to which an item possesses a given quality attribute **2.** function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given quality attribute

3.3281

quality metrics

1. a description of a project or product attribute and how to measure it [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. quality measure

3.3282

quality model

1. defined set of characteristics and of relationships between them, which provides a framework for specifying quality requirements and evaluating quality [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.27*]

3.3283

quality of service

1. set of quality requirements on the collective behavior of one or more objects [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.2*]

3.3284

quality policy

1. a policy specific to the Project Quality Management Knowledge Area, it establishes the basic principles that should govern the organization's actions as it implements its system for quality management [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3285

quality property

1. measurable component of quality [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.3.9*]

3.3286

quality record report

1. report which is generated through dynamic test execution and code analysis to record test results and other output [*ISO/IEC 30130:2016 Software engineering — Capabilities of software testing tools*]

Note 1 to entry: including Test Result, Static Code Analysis Report, Test Incident Report, and Metrics.

3.3287

quality requirement

1. requirement that a software attribute be present in software to satisfy a contract, standard, specification, or other formally imposed document [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.21*] 2. capability of a product to satisfy the stated and implied needs when used under specific conditions [*ISO/IEC TR 24766:2009 Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities, 3.5*] 3. a condition or capability that will be used to assess conformance by validating the acceptability of an attribute for the quality of a result [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. non-functional requirement

3.3288

quality subfactor

1. decomposition of a quality factor or quality subfactor to its technical components [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology, 2.22*]

3.3289

quantitative risk analysis

1. numerical analysis of the effect on overall project objectives of identified risks

3.3290

quantitative risk analysis and modeling techniques

1. commonly used techniques for both event-oriented and project-oriented analysis approaches [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3291

query language

1. language used to access information stored in a database

cf. programming language, specification language

3.3292

questionnaires and surveys

1. written sets of questions designed to quickly accumulate information from a large number of respondents [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3293

queue

1. list in which items are appended to the last position of the list and retrieved from the first position of the list

3.3294

quiescing

1. process of bringing a device or system to a halt by rejecting new requests for work

3.3295

R&D

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

3.3296

R&M

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

3.3297

RACI

1. responsible, accountable, consult, and inform **2.** a common type of responsibility assignment matrix that uses responsible, accountable, consult, and inform statuses to define the involvement of stakeholders in project activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3298

RAM

1. responsibility assignment matrix [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** random-access memory

3.3299

RAM-C

1. reliability, availability, maintainability, and cost [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.3300

random failure

1. failure whose occurrence is unpredictable except in a probabilistic or statistical sense
cf. intermittent fault, transient error

3.3301

random-access memory (RAM)

1. volatile semiconductor storage device which allows data to be written or accessed in approximately the same amount of time, regardless of the data's physical location

Note 1 to entry: often used for caches or main memory in a computer and embedded in an MCU chip. Compare to a CD where data is stored and accessed sequentially.

3.3302

rapid prototyping

1. type of prototyping in which emphasis is placed on developing prototypes early in the development process to permit early feedback and analysis in support of the development process
cf. waterfall model, data structure-centered design, incremental development, input-process-output, modular decomposition

3.3303

rate-monotonic algorithm

1. real-time scheduling algorithm that assigns higher priorities to tasks with shorter periods

3.3304

rating

1. action of mapping the measured value to the appropriate rating level [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.28*]

Note 1 to entry: used to determine the rating level associated with the software for a specific quality characteristic. Rating and rating levels can be applied to characteristics other than quality characteristics.

3.3305

rating interval

- 1.** time interval of the measurement procedure from the time the SUT reaches a stable state of operation to the time the measurement results are fulfilling the required statistical significance [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.13*]

3.3306

rating level

- 1.** scale point on an ordinal scale, which is used to categorize a measurement scale [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.46*]

Note 1 to entry: The rating level enables software product to be classified (rated) in accordance with the stated or implied needs. Appropriate rating levels can be associated with the different views of quality, i.e., Users', Managers', or Developers.'

3.3307

ratio scale

- 1.** scale in which the measurement values have equal distances corresponding to equal quantities of the attribute where the value of zero corresponds to none of the attribute

cf. interval scale, nominal scale, ordinal scale

Note 1 to entry: For example, the size of a software component in terms of LOC is a ratio scale, because the value of zero corresponds to no lines of code and each additional increment represents equal amounts of code.

3.3308

RBS

- 1.** risk breakdown structure [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** resource breakdown structure

3.3309

RDA

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

3.3310

RDBMS

- 1.** Relational Database Management System [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data, 5*]

3.3311

RDF

- 1.** Resource Description Framework [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.3312

RDN

- 1.** Relative Distinguished Name [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service, 4*]

3.3313

reachability graph

- 1.** directed graph of nodes and edges, where the nodes correspond to reachable markings, and the edges correspond to transition occurrences [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.21*]

3.3314

reachability set

- 1.** set of reachable markings of the net, including the initial marking [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.14.5*]

3.3315

reachable marking

1. marking of the net that can be reached from the initial marking by the occurrence of transitions [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.14.4]

3.3316

reactivation

1. cloning a cluster following its deactivation [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.26]

3.3317

Read

read type

1. to access data from a storage device or data medium **2.** data movement type that moves a data group from persistent storage within reach of the functional process which requires it [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method*, 2.23]

cf. destructive read, nondestructive read, write

Note 1 to entry: A read is considered to include certain associated data manipulations necessary to achieve the read.

3.3318

read-only

1. property that causes no state changes [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.155]

Note 1 to entry: That is, it does no updates.

3.3319

read-only memory (ROM)

1. non-volatile semiconductor storage device, from which data cannot be removed once it is written

3.3320

readability

1. ease with which a system's source code can be read and understood, especially at the detailed, statement level

3.3321

reader note

1. comment made by a reader about a diagram and placed on the diagram page [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.97]

Note 1 to entry: A reader note is not part of the diagram itself, but rather is used for communication about a diagram during model development.

3.3322

reading reference

1. data storage entity or record, or interface record from another software or system containing data retrieved in a BFC [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*, 3.8]

Note 1 to entry: The number of reading references is equal to 0 for all BFC types where it is applicable.

3.3323

ready to use software product (RUSP)

1. software product available to any user, at cost or not, to use without the need to conduct development activities [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.6]
cf. commercial off the shelf (COTS)

Note 1 to entry: includes the product description (including cover information, data sheet, and website information; the user documentation necessary to install and use the software, including any configuration of the operating system or target computer required to operate the product; the software contained on a computer sensible media (e.g., disk or CD-ROM) or

367

internet downloadable. Includes software produced and supported without typical commercial fees and licensing considerations.

3.3324

real address

- 1.** address of a storage location in the main storage part of a virtual storage system
cf. virtual address

3.3325

real storage

- 1.** main storage portion of a virtual storage system
cf. virtual storage

3.3326

real type

- 1.** data type whose members can assume real numbers as values and can be operated on by real number arithmetic operations, such as addition, subtraction, multiplication, division, and square root
cf. character type, enumeration type, integer type, logical type

3.3327

real-time

- 1.** problem, system, or application that is concurrent and has timing constraints whereby incoming events must be processed within a given timeframe **2.** pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process

3.3328

real-time clock

- 1.** integrated circuit that tracks the current time in human units

3.3329

real-time clock (RTC)

- 1.** integrated circuit that tracks the current time in human units

3.3330

real-time operating system (RTOS)

- 1.** operating system intended to handle transaction requests immediately upon receipt

3.3331

real-time scheduling theory

- 1.** theory for priority-based scheduling of concurrent tasks with hard deadlines

Note 1 to entry: It addresses how to determine whether a group of tasks, whose individual CPU utilization is known, will meet their deadlines.

3.3332

real-world object

- 1.** entity that exists in a three-dimensional form and, by association, implies similar properties or behavior to software functions *[ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.41]*

EXAMPLE: printer, filing cabinet, file folder, and sheet of paper

3.3333

realization

- 1.** representation of interface responsibilities through specified algorithms and representation properties *[IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.156]*

Note 1 to entry: The realization states "how" a responsibility is met; it is the statement of the responsibility's method. Realization consists of any necessary representation properties together with the algorithm (if any). A realization can involve representation properties or an algorithm, or both. For example, an attribute typically has only a representation and no

algorithm. An algorithm that is a "pure algorithm" (i.e., without any representation properties) uses only literals; it does not "get" any values as its inputs. Finally, a derived attribute or operation typically has both an algorithm and representation properties.

3.3334

receptacle

1. operation interface in which a computational component plays a client role [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 7.1.16] **2.** named connection point that describes the component's ability to use a reference supplied by some external agent [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

3.3335

recommendation

1. provision that conveys advice or guidance [*ISO/IEC 14143-2:2011 Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1*, 3.9]

[SOURCE: ISO/IEC Guide 2:2004]

3.3336

recommended element

1. element that is not required to be present in a tag but is strongly encouraged to be included by a tag creator

3.3337

record

1. set of related data items treated as a unit [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.20] **2.** document stating results achieved or providing evidence of activities performed

EXAMPLE: In stock control, the data for each invoice could constitute one record. Audit reports, incident reports, training records or minutes of meetings.

3.3338

record element type (RET)

1. user-recognizable sub-group of data element types within a data function [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.46]

3.3339

record type

1. an entity type in a logical file [*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*]

3.3340

records management system

1. a specific set of processes, related control functions, and tools that are consolidated and combined to record and retain information about the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3341

recoverability

1. degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.5.4] **2.** degree to which object state changes resulting from failed transactions are cancelled [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 13.7.1.4] cf. survivability

3.3342

recovery

1. restoration of a system, program, database, or other system resource to a state in which it can perform required functions **2.** cloning a cluster after cluster failure or deletion [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 8.1.25*]
cf. backward recovery, checkpoint, forward recovery

3.3343

recursion

1. process in which a software module calls itself **2.** process of defining or generating a process or data structure in terms of itself

cf. simultaneous recursion

3.3344

recursive

1. pertaining to a software module that calls itself **2.** pertaining to a process or data structure that is defined or generated in terms of itself

cf. reflexive

3.3345

redundancy

1. in fault tolerance, the presence of auxiliary components in a system to perform the same or similar functions as other elements for the purpose of preventing or recovering from failures

cf. active redundancy, diversity, homogeneous redundancy, standby redundancy

3.3346

reengineering

1. examination and alteration of software to reconstitute it in a new form, including the subsequent implementation of the new form [*ISO/IEC TR 19759:2016 Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK), 5.4.2*] **2.** complete cycle of performing reverse engineering followed by forward engineering

3.3347

reentrant

reenterable

1. pertaining to a software module that can be entered as part of one process while also in execution as part of another process and still achieve the desired results

3.3348

reentry point

1. place in a software module at which the module is reentered following a call to another module

3.3349

refactor

1. to restructure software code without altering its behavior for the purpose of improving quality attributes, easing future extension or adaptation, or adhering to an architectural style [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3350

reference expression

1. expression that uniquely identifies a box, a node or function, a diagram, or a model page within an IDEF0 model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.98*]

3.3351

reference FSM method

1. an FSM method to be used for comparison reasons when verifying the functional size measurement results [*ISO/IEC TR 14143-4:2002 Information technology — Software measurement — Functional size measurement — Part 4: Reference model, 3.3*]

3.3352

reference mode

1. usage mode that is intended to provide quick access to specific information for software users who are generally familiar with the software functions [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.42*]

3.3353

reference node

1. node of a Petri net that is a representative of another node, possibly defined on another page of the net graph [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.14*]

3.3354

reference place

1. reference node that represents a place and refers to either another reference place or to a place [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.15*]

3.3355

reference point

1. interaction point defined in an architecture for selection as a conformance point in a specification which is compliant with that architecture [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 10.6*]

3.3356

reference transition

1. reference node that represents a transition and refers to either another reference transition to a transition [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.16*]

3.3357

reference user requirement collection (RUR Collection)

1. a subset of RUR which is selected to match the purpose in a specific evaluation [*ISO/IEC TR 14143-4:2002 Information technology — Software measurement — Functional size measurement — Part 4: Reference model, 3.5*]

3.3358

reference user requirements (RUR)

1. a standard set of user requirements which conforms to the requirements [*ISO/IEC TR 14143-4:2002 Information technology — Software measurement — Functional size measurement — Part 4: Reference model, 3.4*]

3.3359

referential integrity

1. guarantee that a reference refers to an object that exists [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.157*] 2. guarantee that all specified conditions for a relationship hold true [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.157*]

EXAMPLE: If a class is declared to require at least one instance of a related state class, it would be invalid to allow an instance that does not have such a relationship.

3.3360

refinement

1. process of transforming one specification into a more detailed specification [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.6*]

3.3361

reflective construct

1. construct that is viewed as the cause of measures in the relationship between a construct and its measures [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks, 3.13*]

3.3362

reflexive

- 1.** in a relationship, the condition when the same data object plays both (binary or many (n-ary)) roles [ISO/IEC 15476-4:2005 *Information technology — CDIF semantic metamodel — Part 4: Data models*, 6.5]
cf. recursive

3.3363

reflexive ancestor (of a class)

- 1.** class itself or any of its generic ancestors [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.158]
cf. generic ancestor

3.3364

refresh

- 1.** method to keep data in volatile memory by rewriting the data before it disappears from the memory

Note 1 to entry: needed for memory with a circuit architecture in single stable state

3.3365

regid

- 1.** registration identifier [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.27]

3.3366

register

- 1.** small storage unit in a processing unit

Note 1 to entry: Registers can be set up in CPU, microcontroller, digital signal processor, or microprocessor.

3.3367

register bank

- 1.** group of registers in a microprocessor chip

3.3368

registration identifier

regid

- 1.** unique identifier for an entity [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.27]

3.3369

regression analysis

- 1.** an analytic technique where a series of input variables are examined in relation to their corresponding output results in order to develop a mathematical or statistical relationship [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3370

regression test

- 1.** retesting to detect faults introduced by modification

3.3371

regression testing

- 1.** selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements **2.** testing required to determine that a change to a system component has not adversely affected functionality, reliability or performance and has not introduced additional defects [ISO/IEC 90003:2014 *Software engineering — Guidelines for the application of ISO 9001:2008 to computer software*, 3.11] **3.** testing following modifications to a test item or to its operational environment, to identify whether regression failures occur [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.32]

3.3372

regulation

1. requirements imposed by a governmental body. These requirements can establish product, process or service characteristics, including applicable administrative provisions that have government-mandated compliance [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.3373

relation

1. set of relationships of the same relationship type [ISO/IEC 14769:2001 *Information technology — Open Distributed Processing — Type Repository Function*, 3.2.3] **2.** association between two or more domains of entities [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 8.13]

3.3374

relational database management system

1. management system for relational database [ISO/IEC 25024:2015 *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.34]

Note 1 to entry: In order to use relational data base management systems (RDBMS), it is necessary to represent relational model of data that organizes data with specific characteristics (tables or relations, unique key, etc.)

3.3375

relationship

1. real-world association among one or more entities [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2] **2.** association between two (not necessarily distinct) classes that is deemed relevant within a particular scope and purpose [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.159] **3.** semantic connection between model elements **4.** association between two or more entities [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 8.14] **5.** predicate involving two or more roles with assigned values [ISO/IEC 14769:2001 *Information technology — Open Distributed Processing — Type Repository Function*, 3.2.1]

Note 1 to entry: The association is named for the sense in which the instances are related. A relationship can be represented as a time-varying binary relation between the instances of the current extents of two state classes.

3.3376

relationship instance

1. association of specific instances of the related classes [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.160]

3.3377

relationship name

1. verb or verb phrase that reflects the meaning of the relationship expressed between the two entities shown on the diagram on which the name appears [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.161]

Note 1 to entry: [key style]

3.3378

relationship type

1. type of relationship which expresses the number and type of the roles [ISO/IEC 14769:2001 *Information technology — Open Distributed Processing — Type Repository Function*, 3.2.2]

3.3379

relative address

1. address that must be adjusted by the addition of an offset to determine the address of the storage location to be accessed

cf. absolute address, base address, indexed address, self-relative address

3.3380

relative chain frequency

1. relative frequency of using the l-th chain type by an emulated user of the i-th type [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.14]

3.3381

RELAX NG

1. regular language description for XML/New Generation [ISO/IEC 15909-2:2011 *Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.2.6]

3.3382

release

1. particular version of a configuration item that is made available for a specific purpose [IEEE 828-2012 *IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.12; ISO/IEC 90003:2014 *Software engineering — Guidelines for the application of ISO 9001:2008 to computer software*, 3.10] 2. collection of new or changed configuration items that are tested and introduced into a live environment together [IEEE 828-2012 *IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1; ISO/IEC 19770-1:2012 *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3.12] 3. collection of one or more new or changed configuration items deployed into the live environment as a result of one or more changes [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.28] 4. software version that is made formally available to a wider community [IEEE 828-2012 *IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] 5. delivered version of an application which includes all or part of an application [IEEE 828-2012 *IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] 6. set of grouped change requests, established in the Application Change Management Process, which are designed, developed, tested, and deployed as a cohesive whole [ISO/IEC 16350-2015 *Information technology — Systems and software engineering — Application management*, 4.28]
cf. version

EXAMPLE: source code, code for execution, or multiple software assets packaged into an internal production release and tested for a target platform

Note 1 to entry: Release management includes defining acceptable quality levels for release, authority to authorize the release, and release procedures.

3.3383

release engineer

1. person responsible for coordinating development toward a release

Note 1 to entry: The release engineer will monitor pending issues for a given release, oversee the code freeze, and tag the release once it gets out the door.

3.3384

release map

1. a displayed forecast of when software features will be released and how they will be grouped into releases [Software Extension to the PMBOK® Guide Fifth Edition]

3.3385

release plan

1. plan that describes what portions of system functionality will be implemented in which releases and the rationale for each release [IEEE 828-2012 *IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1]

Note 1 to entry: It includes or provides reference to a description of release contents, release schedule, release impacts, and release notifications.

3.3386

relevant stakeholder

1. stakeholder that is identified for involvement in specified activities and is included in a plan

3.3387

reliability

- 1.** ability of a system or component to perform its required functions under stated conditions for a specified period of time **2.** degree to which a system, product or component performs specified functions under specified conditions for a specified period of time [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.5] **3.** degree to which an object or an object's services provide agreed or expected functionality during a defined time period under specified conditions [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.29]
cf. availability, MTBF

Note 1 to entry: Dependability characteristics include availability and its inherent or external influencing factors, such as availability, reliability (including fault tolerance and recoverability), security (including confidentiality and integrity), maintainability, durability, and maintenance support. Wear or aging does not occur in software. Limitations in reliability are due to faults in requirements, design, and implementation, or due to contextual changes.

3.3388

reliability growth

- 1.** improvement in reliability that results from correction of faults

3.3389

reliability testing

- 1.** type of testing conducted to evaluate the ability of a test item to perform its required functions, including evaluating the frequency with which failures occur, when it is used under stated conditions for a specified period of time [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.33]

3.3390

relocatable

- 1.** pertaining to code that can be loaded into any part of main memory
cf. relocating loader

Note 1 to entry: The starting address is established by the loader, which then adjusts the addresses in the code to reflect the storage locations into which the code has been loaded.

3.3391

relocatable address

- 1.** address that is to be adjusted by the loader when the computer program containing the address is loaded into memory
cf. absolute address

3.3392

relocatable code

- 1.** code containing addresses that are to be adjusted by the loader to reflect the storage locations into which the code is loaded
cf. absolute code

3.3393

relocate

- 1.** to move machine code from one portion of main memory to another and to adjust the addresses so that the code can be executed in its new location

3.3394

relocating assembler

- 1.** assembler that produces relocatable code
cf. absolute assembler

3.3395

relocating loader

relative loader

1. loader that reads relocatable code into main memory and adjusts the addresses in the code to reflect the storage locations into which the code has been loaded

cf. absolute loader

3.3396

relocation dictionary

1. part of an object module or load module that identifies the addresses that must be adjusted when a relocation occurs

3.3397

relocation transparency

1. distribution transparency which masks relocation of an interface from other interfaces bound to it [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 4.4.1.5]

3.3398

relocator

1. object that manages a repository of locations for interfaces, including locations of management functions for the cluster supporting those interfaces [ISO/IEC 10746-3:2009 *Information technology — Open Distributed Processing — Reference Model: Architecture*, 14.3.1.1]

3.3399

remote job entry (RJE)

remote batch entry

1. submission of jobs through a remote input device connected to a computer through a data link

3.3400

remote terminal emulator (RTE)

1. data processing system realizing a set of emulated users [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.15]

3.3401

repair

defect removal

1. correction of defects that have resulted from errors in external design, internal design, or code

EXAMPLE: missing functions that do not result in application failure (external design error) or errors resulting in a stop-run situation (code error)

3.3402

reparent

1. to move the changes developed under one branch into another

Note 1 to entry: The changes are not committed to the original branch.

3.3403

repeatability (of results of measurements)

1. closeness of the agreement between the results of successive measurements of the same measurand carried out under the same conditions of measurement [ISO/IEC 25021:2012 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*, 4.15; ISO/IEC TR 14143-3:2003 *Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*, 3.8]

Note 1 to entry: These conditions are called repeatability conditions. Repeatability conditions include the same measurement procedure, the same observer, the same measuring instrument, used under the same conditions; the same location; repetition over a short period of time. Repeatability is expressed quantitatively in terms of the dispersion characteristics of the results.

3.3404

repetitive addressing

- 1.** method of implied addressing in which the operation field of a computer instruction is understood to address the operands of the last instruction executed

cf. one-ahead addressing

3.3405

replaceability

- 1.** degree to which a product can replace another specified software product for the same purpose in the same environment [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.8.3*]

cf. adaptability, installability

Note 1 to entry: Replaceability of a new version of a software product is important to the user when upgrading. Replaceability will reduce lock-in risk, so that other software products can be used in place of the present one,

3.3406

replication

- 1.** copying a software product from one medium to another [*ISO/IEC 90003:2014 Software engineering — Guidelines for the application of ISO 9001:2008 to computer software, 3.13*]

3.3407

replication schema

- 1.** specification of constraints on the replication of an object including both constraints on the availability of the object and constraints on the performance of the object [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 16.7.1.1*]

3.3408

replication transparency

- 1.** distribution transparency which masks the use of a group of mutually behaviorally compatible objects to support an interface [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 4.4.1.6*]

Note 1 to entry: Replication is often used to enhance performance and availability.

3.3409

repo bloat

- 1.** changes recorded in the repository that do not contribute anything useful to the project's history

EXAMPLE: a large erroneous commit and its associated back-out operation

Note 1 to entry: A repo master can reduce repo bloat through repo surgery.

3.3410

repo master

- 1.** person in charge of the version control system's repository

Note 1 to entry: one who has rights to perform repo surgery.

3.3411

repo surgery

- 1.** changes made directly to the version control system's repository, bypassing the system's commands

EXAMPLE: by issuing database commands or directly manipulating system files

Note 1 to entry: Through repo surgery, a repo master can perform operations that the version control system does not directly support.

3.3412

report

- 1.** information item that describes the results of activities such as investigations, observations, assessments, or tests [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.21] **2.** an output of data in a layout specified by the user [*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*]

Note 1 to entry: The output medium used is not relevant for FPA and it can pertain to both an external output and an external inquiry.

3.3413

report standard

- 1.** standard that describes the characteristics of describing results of engineering and management activities

3.3414

reporting systems

- 1.** facilities, processes and procedures used to generate or consolidate reports from one or more information management systems and facilitate report distribution to the project stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3415

repository

- 1.** collection of all system element or software-related artifacts belonging to a system [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.49] **2.** location/format in which such a collection is stored [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1] **3.** organized and persistent data storage that allows data retrieval [*ISO/IEC 29155-1:2011 Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.9] **4.** collection of all software-related artifacts belonging to a system [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1]

Note 1 to entry: Artifacts are, for example, the software engineering environment. Benchmarking repository is a repository which is designated for use as the source of comparative measures for the purpose of benchmarking.

3.3416

repository administrator

- 1.** person or organization that maintains and administrates data in a repository [*ISO/IEC 29155-1:2011 Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*, 2.10]

3.3417

repository owner

- 1.** person or organization that owns and maintains a benchmarking repository [*ISO/IEC 29155-2:2013: Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*, 3.2]

3.3418

representation

- 1.** logical portrayal of a physical, operational, or conceptual image or situation **2.** one or more properties used by an algorithm for the realization of a responsibility [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.162]

EXAMPLE: picture, drawing, block diagram, description, icon, symbol

3.3419

representation property

- 1.** property on which an algorithm operates [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.163]

3.3420

representation standard

- 1.** standard that describes the characteristics of portraying aspects of an engineering or management product

3.3421

reproducibility (of results of measurements)

- 1.** closeness of the agreement between the results of measurements of the same measurand carried out under changed conditions of measurement [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*, 4.16; *ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*, 3.9]

Note 1 to entry: A valid statement of reproducibility requires specification of the conditions changed. The changed conditions include the principle of measurement; method of measurement; observer; measuring instrument; reference standard; location; conditions of use; time. Reproducibility can be expressed quantitatively in terms of the dispersion characteristics of the results. Results are here usually understood to be corrected results.

3.3422

request

- 1.** a message sent from one object (the sender) to another object (the receiver), directing the receiver to fulfill one of its responsibilities [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.164] **2.** message issued by a client to cause a service to be performed [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.12] **3.** information item that initiates a defined course of action or change to fulfill a need [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.22]

cf. message

Note 1 to entry: Specifically, a request can be for the value of an attribute, for the value of a participant property, for the application of an operation, or for the truth of a constraint. Request also encompasses sentences of such requests. Logical sentences about the property values and constraints of objects are used for queries, pre-conditions, post-conditions, and responsibility realizations.

3.3423

request for change

- 1.** proposal for a change to be made to a system, service, component, or the service management system **2.** proposal for a functional or non-functional change to be made to an existing application [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.30]
cf. change request, modification request.

Note 1 to entry: A change to a service includes the provision of a new service or the removal of a service which is no longer required.

3.3424

request for information (RFI)

- 1.** a type of procurement document whereby the buyer requests a potential seller to provide various pieces of information related to a product or service or seller capability [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3425

request for proposal (RFP)

request for tender

solicitation package

- 1.** collection of formal documents that includes a description of the desired form of response from a potential supplier, the relevant statement of work for the supplier, and required provisions in the supplier agreement **2.** a type of procurement document used to request proposals from prospective sellers of products or services. In some application areas, it may have a narrower or more specific meaning. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3426

request for quotation (RFQ)

1. a type of procurement document used to request price quotations from prospective sellers of common or standard products or services. Sometimes used in place of request for proposal and in some application areas, it may have a narrower or more specific meaning. [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition]

3.3427

request form

1. description or pattern that can be evaluated or performed multiple times to cause the issuing of requests [ISO/IEC 19500-1:2012 *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.3.2]

3.3428

requested change

1. [a formally documented change request that is submitted for approval to the integrated change control process [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition]

3.3429

required inputs

1. set of items necessary to perform the minimum verification and validation (V&V) tasks mandated within any life cycle activity [IEEE 1012-2012 *IEEE Standard for System and Software Verification and Validation*, 3.1]

3.3430

required outputs

1. set of items produced as a result of performing the minimum verification and validation tasks mandated within any life cycle activity [IEEE 1012-2012 *IEEE Standard for System and Software Verification and Validation*, 3.1]

3.3431

requirement

1. statement that translates or expresses a need and its associated constraints and conditions [ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.41; ISO/IEC/IEEE 29148:2011 *Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.19] **2.** condition or capability that must be met or possessed by a system, system component, product, or service to satisfy an agreement, standard, specification, or other formally imposed documents [IEEE 730-2014 *IEEE Standard for Software Quality Assurance Processes*, 3.2] **3.** provision that contains criteria to be fulfilled [ISO/IEC 14143-2:2011 *Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1*, 3.10] **4.** a condition or capability that must be present in a product, service, or result to satisfy a contract or other formally imposed specification [A *Guide to the Project Management Body of Knowledge (PMBOK® Guide)* — Fifth Edition] cf. design requirement, functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement

EXAMPLE: software component requirement

Note 1 to entry: Requirements exist at different tiers and express the need in high-level form. A requirement is denoted by the word 'shall' and when used includes both the exclusive and applicable optional requirements. Requirements provide value when delivered, satisfied, or met. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders.

3.3432

requirement standard

1. a standard that describes the characteristics of a requirements specification

3.3433

requirements allocation

1. assignment or budgeting of top-level functional or nonfunctional requirements among the lower-level partitioned functions for accomplishment

Note 1 to entry: In this manner, the system elements that perform all or part of specific requirements are identified.

3.3434

requirements analysis

1. process of studying user needs to arrive at a definition of system, hardware, or software requirements **2.** process of studying and refining system, hardware, or software requirements **3.** systematic investigation of user requirements to arrive at a definition of a system [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **4.** determination of product- or service-specific performance and functional characteristics based on analyses of customer needs, expectations, and constraints; operational concept; projected utilization environments for people, products, services, and processes; and measures of effectiveness

3.3435

requirements attributes

1. set of properties associated with requirements [*ISO/IEC TR 24766:2009 Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities*, 3.7]

3.3436

requirements derivation

1. changing or translation of a requirement through analysis into a form that is suitable for low-level analysis or design **2.** in a hierarchical structure, the next lower level that is associated with a given element

3.3437

requirements document

1. document containing any combination of requirements or regulations to be met by a ready to use software product (RUSP) [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.13]

cf. requirements documentation

EXAMPLE: a technical or ergonomic standard, a requirements list (or model requirements specification) from a group (e.g. a market sector, technical or user association), a law or a decree

3.3438

requirements documentation

1. a description of how individual requirements meet the business need for the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3439

requirements elicitation

1. process through which the acquirer and the suppliers of a system discover, review, articulate, understand, and document the requirements on the system and the life cycle processes [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.18] **2.** use of systematic techniques, such as prototypes and structured surveys, to proactively identify and document customer and end-user needs

3.3440

requirements engineering

1. interdisciplinary function that mediates between the domains of the acquirer and supplier to establish and maintain the requirements to be met by the system, software or service of interest [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.19] *cf.* software requirements engineering

Note 1 to entry: Requirements engineering is concerned with discovering, eliciting, developing, analyzing, determining verification methods, validating, communicating, documenting, and managing requirements

3.3441

requirements flow-down

1. systematic decomposition of system requirements into allocated and derived requirements, appropriately assigned to low-level functional components

3.3442

requirements management

1. activities that ensure requirements are identified, documented, maintained, communicated and traced throughout the life cycle of a system, product, or service [*ISO/IEC/IEEE 29148:2011 Systems and software*

engineering — Life cycle processes — Requirements engineering, 4.1.20] **2.** provision of storing and editing capabilities, tracking history of edition, versioning, author identification, change management, time stamping, user notification for content changes, security rights control [*ISO/IEC TR 24766:2009 Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities, 3.3*] *cf. software requirements management*

3.3443

requirements management plan

1. a component of the project management plan that describes how requirements will be analyzed, documented, and managed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3444

requirements partitioning

1. separation or decomposing of a top-level requirement or design into successively lower-level detailed requirements or design

cf. decomposition

3.3445

requirements phase

1. period of time in the software life cycle during which the requirements for a software product are defined and documented

3.3446

requirements review

1. process or meeting during which the requirements for a system, hardware item, or software item are presented to project personnel, managers, users, customers, or other interested parties for comment or approval

cf. code review, design review, formal qualification review, test readiness review

Note 1 to entry: Types include system requirements review, software requirements review.

3.3447

requirements specification

1. document that specifies the requirements for a system or component

cf. design description, functional specification, performance specification

Note 1 to entry: Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.

3.3448

requirements specification language

1. specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document hardware or software requirements

cf. design language

3.3449

requirements traceability

1. identification and documentation of the derivation path (upward) and allocation/ flow-down path (downward) of requirements in the requirements hierarchy **2.** discernible association between a requirement and related requirements, implementations, and verifications **3.** traceabilities in domain and application requirements respectively and those between them [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering, 3.18*]

3.3450

requirements traceability matrix (RTM)

1. table that links requirements to their origin and traces them throughout the project life cycle **2.** a grid that links product requirements from their origin to the deliverables that satisfy them [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3451

requirements traceability tool

- 1.** software development tool that establishes a traceability among itemized software requirements specifications, design elements, code elements, and test cases

Note 1 to entry: It also supports various associated query, analysis, and report-generation capabilities.

3.3452

requirements validation

- 1.** confirmation by examination that requirements (individually and as a set) define the right system as intended by the stakeholders [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.22]

cf. requirements verification, software requirements validation

3.3453

requirements verification

- 1.** confirmation by examination that requirements (individually and as a set) are well formed [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1.23]

cf. requirements validation, software requirements verification

Note 1 to entry: This means that a requirement or a set of requirements has been reviewed to ensure the characteristics of good requirements are achieved.

3.3454

reseller

- 1.** organization that purchases goods or services with an intention of selling them to another customer and possibly supporting them. [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.29]

3.3455

reserve

contingency allowance

- 1.** a provision in the project management plan to mitigate cost and/or schedule risk. Often used with a modifier (e.g., management reserve, contingency reserve) to provide further detail on what types of risk are meant to be mitigated [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. buffer

3.3456

reserve analysis

- 1.** an analytical technique to determine the essential features and relationships of components in the project management plan to establish a reserve for the schedule duration, budget, estimated cost, or funds for a project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3457

reserved word

- 1.** word in a programming language, of which the meaning is fixed by the rules of that language and which, in certain or all contexts, cannot be used by the programmer for any purpose other than its intended one

EXAMPLE: IF, THEN, WHILE

3.3458

reset

- 1.** to set a variable, register, or other storage location back to a prescribed state
cf. clear, initialize

3.3459

residual control

- 1.** microprogramming technique in which the meaning of a field in a microinstruction depends on the value in an auxiliary register

cf. bit steering, two-level encoding

3.3460

residual risk

1. a risk that remains after risk responses have been implemented [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** risk remaining after risk treatment [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.15]

3.3461

resource

1. asset that is utilized or consumed during the execution of a process [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.37; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.42; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.38] **2.** skilled human resources (specific disciplines either individually or in crews or teams), equipment, services, supplies, commodities, materiel, budgets, or funds [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **3.** role (with respect to that action) in which the enterprise object fulfilling the role is essential to the action, requires allocation, or can become unavailable [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language*, 6.3.5] **4.** any physical or virtual component of limited availability within a computer system available for a given purpose and managed by the runtime platform [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

EXAMPLE: diverse entities such as funding, personnel, facilities, capital equipment, tools, and utilities such as power, water, fuel and communication infrastructures

Note 1 to entry: Allocation of a resource can constrain other behaviors for which that resource is essential. Resources can be reusable, renewable or consumable. A consumable resource can become unavailable after some amount of use or after some amount of time (in case a duration or expiry has been specified for the resource).

3.3462

resource allocation

1. allocation (partitioning) of responsibility to different organizational units

3.3463

resource breakdown structure (RBS)

1. a hierarchical representation of resources by category and type [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3464

resource calendar

1. a calendar that identifies the working days and shifts on which each specific resource is available [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. project calendar

3.3465

resource histogram

1. a bar chart showing the amount of time that a resource is scheduled to work over a series of time periods. Resource availability may be depicted as a line for comparison purposes. Contrasting bars may show actual amounts of resource used as the project progresses [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3466

resource leveling

1. a technique in which start and finish dates are adjusted based on resource constraints with the goal of balancing demand for resources with the available supply [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3467

resource management

1. identification, estimation, allocation, and monitoring of the means used to develop a product or perform a service

3.3468

resource monitor task

- 1.** task ensuring sequential access to a resource

3.3469

resource optimization techniques

- 1.** a technique that is used to adjust the start and finish dates of activities that adjusts planned resource use to be equal to or less than resource availability [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3470

resource smoothing

- 1.** a technique which adjusts the activities of a schedule model such that the requirement for resources on the project does not exceed certain predefined resource limits [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3471

resource utilization

- 1.** degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.2.2*] *cf.* efficiency

Note 1 to entry: Human resources are included as part of efficiency.

3.3472

resource-limited schedule

- 1.** project schedule whose schedule activity, scheduled start dates and scheduled finish dates reflect expected resource availability

3.3473

respecialize

- 1.** change by an instance from being an instance of its current subclass to being an instance of one of the other subclasses in its current cluster [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.165*] *cf.* specialize, unspecialize

3.3474

responding object

- 1.** object taking part in a communication, which is not the initiating object [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.4.2*]

3.3475

response

- 1.** reaction of the unit to a specific stimulus, depending on the current conditions on the unit's lifeline [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.24*]

Note 1 to entry: A response might be the occurrence of internally controlled interaction(s) affecting the environment (external response), or it might be the occurrence of an internally controlled change to the unit's state (internal response).

3.3476

response time

- 1.** elapsed time between the end of an inquiry or command to an interactive computer system and the beginning of the system's response

cf. port-to-port time, think time, turnaround time

3.3477

responsibility

- 1.** generalization of properties (attributes, participant properties, and operations) and constraints [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.166*] **2.** an assignment that can be delegated within a project management plan such that the assigned resource

incurs a duty to perform the requirements of the assignment [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: An instance possesses knowledge, exhibits behavior, and obeys rules. These are collectively referred to as the instance's responsibilities. A class abstracts the responsibilities in common to its instances. A responsibility can apply to each instance of the class (instance-level) or to the class as a whole (class-level).

3.3478

responsibility assignment matrix (RAM)

1. a grid that shows the project resources assigned to each work package [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3479

responsive web design (RWD)

1. method for web page construction to detect the user's screen size and orientation and dynamically change the layout accordingly [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.22*]

3.3480

restart

1. to cause a computer program to resume execution after a failure, using status and results recorded at a checkpoint

3.3481

restart point

rescue point

1. point in a computer program at which execution can be restarted following a failure

3.3482

result

1. information returned to the client [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 3.2.13*] **2.** an output from performing project management processes and activities. Results include outcomes (e.g., integrated systems, revised process, restructured organization, tests, trained personnel, etc.) and documents (e.g., policies, plans, studies, procedures, specifications, reports, etc.) [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. product, deliverable

3.3483

RET

1. record element type [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 4; ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 4*]

3.3484

retainage

1. portion of a contract payment that is withheld until contract completion to ensure full performance of the contract terms

3.3485

retesting

confirmation testing

1. re-execution of test cases that previously returned a "fail" result, to evaluate the effectiveness of intervening corrective actions [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.34*]

3.3486

retirement

1. withdrawal of active support by the operation and maintenance organization, partial or total replacement by a new system, or installation of an upgraded system [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.38; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle*

management — Part 1: Guide for life cycle management, 2.43; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.39] **2.** permanent removal of a system or component from its operational environment

3.3487

retirement phase

- 1.** period of time in the software life cycle during which support for a software product is terminated
cf. software life cycle, system life cycle

3.3488

retrospective meeting

- 1.** a team meeting at the end of an iterative cycle or at the end of a software project to reflect on what went well, what was learned, and what should be done differently next time [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3489

retrospective trace

- 1.** trace produced from historical data recorded during the execution of a computer program
cf. execution trace, subroutine trace, symbolic trace, variable trace

Note 1 to entry: This differs from an ordinary trace, which is produced cumulatively during program execution.

3.3490

return

- 1.** to transfer control from a software module to the module that called it **2.** to assign a value to a parameter that is accessible by a calling module **3.** computer instruction or process that performs the transfer in (1)
cf. return code

3.3491

return code

- 1.** code used to influence the execution of a calling module following a return from a called module

3.3492

return on investment (ROI)

- 1.** ratio of revenue from output (product or service) to development and production costs, which determines whether an organization benefits from performing an action to produce something

3.3493

return value

- 1.** value assigned to a parameter by a called module for access by the calling module

3.3494

reusability

- 1.** degree to which an asset can be used in more than one system, or in building other assets [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3; ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.7.2*]**2.** in a reuse library, the characteristics of an asset that make it easy to use in different contexts, software systems, or in building different assets [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]
cf. generality

3.3495

reusable

- 1.** pertaining to a software module or other work product that can be used in more than one computer program or software system

3.3496

reusable product

- 1.** system, software, or hardware product developed for one use but having other uses, or one developed specifically to be usable on multiple projects or in multiple roles on one project [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*]

EXAMPLE: commercial off-the-shelf (COTS) products, acquirer-furnished products, products in reuse libraries, and preexisting developer products

Note 1 to entry: Each use can include all or part of the product and can involve its modification. This term can be applied to any software or system product (for example, requirements or architectures), not just to software or system itself.

3.3497

reuse

1. use of an asset in the solution of different problems [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*] **2.** building a software system at least partly from existing pieces to perform a new application

3.3498

reuse sponsor

1. member of the organization's management who authorized, approves, promotes, and obtains the funding and other resources for the reuse program [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*]

3.3499

reused source statement

1. unmodified source statement obtained for the product from an external source

3.3500

revenue alternative

1. alternative that is described in terms of its complete cash-flow stream with expense and income
cf. service alternative

3.3501

reverse engineering

1. determining what existing software will do and how it is constructed (to make intelligent changes) **2.** a software engineering approach that derives a system's design or requirements from its code

3.3502

reversible execution

backward execution

playback

replay

reverse execution

1. debugging technique in which a history of program execution is recorded and then replayed under the user's control, in either the forward or backward direction

3.3503

review

1. process, which can include a meeting, in which work products are presented to some stakeholders for comment or approval [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*] **2.** process or meeting during which a software product is presented to project personnel, managers, users, customers, user representatives, or other interested parties for comment or approval [*IEEE 1028-2008 IEEE Standard for Software Reviews and Audits, 3.5*] **3.** process or meeting during which a work product, or set of work products, is presented to project personnel, managers, users, customers, or other interested parties for comment or approval [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 3.52*]

3.3504

review/audit output

audit output

review output

1. review or audit artifacts that are expected through conduct of the technical review or audit and that can be considered elements of exit criteria [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1*]

3.3505

revocation

- 1.** process of revoking an entitlement or entitlement schema [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.21]

Note 1 to entry: An entitlement is sometimes revoked by the organization which originally issued it. The entitlement schema (Ent) enables the recording of entitlement revocations. Specific Ent transactions can also be revoked, e.g., to correct errors or record the rescinding of entitlement allocations.

3.3506

rework

- 1.** action taken to bring a defective or nonconforming component into compliance with requirements or specifications [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3507

RFC

- 1.** request for change [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.2]

3.3508

RFI

- 1.** request for information [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3509

RFP

- 1.** request for proposal [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3510

RFQ

- 1.** request for quotation [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3511

right

- 1.** privilege or benefit granted by a software entitlement [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.22]

3.3512

risk

- 1.** effect of uncertainty on objectives [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.16; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.44; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.40] **2.** combination of the probability of an abnormal event or failure and the consequence(s) of that event or failure to a system's components, operators, users, or environment [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.3] **3.** combination of the probability of an event and its consequence [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management*, 3.5] **4.** measure that combines both the likelihood that a system hazard will cause an accident and the severity of that accident [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans*, 3.1.3] **5.** function of the probability of occurrence of a given threat and the potential adverse consequences of that threat's occurrence [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.12] **6.** combination of the probability of occurrence and the consequences of a given future undesirable event [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1] **7.** uncertain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. opportunity

Note 1 to entry: Risk is often characterized by reference to potential events and consequences, or a combination of these. Risk is often expressed in terms of a combination of the consequences of an event and the associated likelihood of occurrence. Uncertainty is the state, even partial, of deficiency of information related to, understanding or knowledge of, an event, its consequence, or likelihood.

[SOURCE: ISO Guide 73:2009]

3.3513

risk action request

1. recommended treatment alternatives and supporting information for one or more risks determined to be above a risk threshold [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.7*]

3.3514

risk analysis

1. process of examining identified risk factors for probability of occurrence, potential loss, and potential risk-handling strategies

3.3515

risk appetite

1. the degree of uncertainty an entity is willing to take on, in anticipation of a reward [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3516

risk audit

1. examination and documentation of the effectiveness of risk responses in dealing with identified risks and their root causes, as well as the effectiveness of the risk management process [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3517

risk avoidance

1. course of action that removes a risk factor from further consideration **2.** a risk response strategy whereby the project team acts to eliminate the threat or protect the project from its impact [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

EXAMPLE: changing the requirements, extending the schedule, or transferring the risk factor to another domain

3.3518

risk breakdown structure (RBS)

1. a hierarchical representation of risks according to their risk categories [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3519

risk categorization

1. organization by sources of risk (e.g., using the RBS), the area of the project affected (e.g., using the WBS), or other useful category (e.g., project phase) to determine the areas of the project most exposed to the effects of uncertainty [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3520

risk category

1. a group of potential causes of risk [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. source

EXAMPLE: technical, legal, organizational, safety, economic, engineering, cost, schedule

Note 1 to entry: A risk category is a characterization of a source of risk.

3.3521

risk criteria

1. terms of reference by which the significance of risk is assessed [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.9*] **2.** terms of reference against which the significance of a risk is evaluated [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.17*]

Note 1 to entry: Risk criteria can include associated cost and benefits, legal and statutory requirements, socio-economic and environmental aspects, the concerns of stakeholders, priorities and other inputs to the assessment.

3.3522

risk data quality assessment

- 1.** technique to evaluate the degree to which the data about risks is useful for risk management [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3523

risk exposure

- 1.** potential loss presented to an individual, project, or organization by a risk [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.10*] **2.** function of the likelihood that the risk will occur and the magnitude of the consequences of its occurrence [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.10*] **3.** product of probability times potential loss for a risk factor

Note 1 to entry: Risk exposure is commonly defined as the product of a probability and the magnitude of a consequence, that is, an expected value or expected exposure.

3.3524

risk factor

- 1.** potential problem that would be detrimental to a planned activity, project, or program, characterized by the probability of problem occurrence ($0 \dots p \dots 1$) and a potential loss (of life, money, property, reputation, and so on) if the problem occurs

Note 1 to entry: Both probability and potential loss might change over time.

3.3525

risk handling

- 1.** course of action taken in response to a risk factor

Note 1 to entry: includes risk acceptance, risk avoidance, risk transfer, and risk mitigation.

3.3526

risk identification

- 1.** organized, systematic approach to determining the risk factors associated with a planned activity, project, or program
cf. identify risks

3.3527

risk leverage factor (rlf)

- 1.** $rlf = (reb - rea)/rmc$, where reb is risk exposure before risk mitigation, rea is risk exposure after risk mitigation, and rmc is the risk mitigation activity's cost

Note 1 to entry: Larger rlfs indicate better mitigation strategies.

3.3528

risk management

- 1.** organized process for identifying and handling risk factors **2.** organized, analytic process to identify what might cause harm or loss (identify risks); to assess and quantify the identified risks; and to develop and, if needed, implement an appropriate approach to prevent or handle causes of risk that could result in significant harm or loss

Note 1 to entry: The primary goal of risk management is to identify and respond to potential problems with sufficient lead-time to avoid a crisis situation. Includes initial identification and handling of risk factors as well as continuous risk management.

3.3529

risk management plan

- 1.** description of how the elements and resources of the risk management process will be implemented within an organization or project [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.11*] **2.** a component of the project, program or portfolio management plan that describes how risk

management activities will be structured and performed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. risk register

3.3530

risk management process

1. continuous process for systematically identifying, analyzing, treating, and monitoring risk throughout the life cycle of a product or service [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.12*]

3.3531

risk management system

1. set of elements of an organization's management system concerned with managing risk [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.13*]

Note 1 to entry: Management system elements can include strategic planning, decision making, and other processes for dealing with risk. The culture of an organization is reflected in its risk management system.

3.3532

risk metric

1. objective measure associated with a risk factor to be mitigated

3.3533

risk mitigation

2. a risk response strategy whereby the project team acts to reduce the probability of occurrence or impact of a risk [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

Note 1 to entry: includes executing contingency plans when a risk metric crosses a predetermined threshold (when a risk becomes an issue or results in a problem)

3.3534

risk monitoring and control

1. tracking identified risks, monitoring residual risks, identifying new risks, executing risk response plans, and evaluating their effectiveness throughout the project life cycle

3.3535

risk profile

1. chronological record of a risk's current and historical risk state information [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.14*]

3.3536

risk reassessment

1. Risk reassessment is the identification of new risks, reassessment of current risks, and the closing of risks that are outdated [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3537

risk reduction

1. reducing the probability and/or potential impact of a risk factor

Note 1 to entry: Risk reduction might involve research, prototyping, and other means of exploration.

3.3538

risk reduction measure

1. steps taken to reduce or mitigate risk [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.18*]

3.3539

risk register

1. a document in which the results of risk analysis and risk response planning are recorded [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. risk management plan

Note 1 to entry: The risk register details all identified risks, including description, category, cause, probability of occurring, impact(s) on objectives, proposed responses, owners, and current status. It can be kept in a database.

3.3540

risk source

1. element that, alone or in combination, has the intrinsic potential to give rise to risk [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.19*]

Note 1 to entry: A hazard in ISO Guide 73:2009 is an instance of a risk source. A fault, an error, or a failure in the context of reliability can be a risk source. A threat in the context of security, a threat agent, or an adverse action can be a risk source.

3.3541

risk state

1. current project risk information relating to an individual risk [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.15*]

Note 1 to entry: The information concerning an individual risk includes the current description, causes, likelihood, consequences, estimation scales, confidence of the estimates, treatment, threshold, and an estimate of when the risk will reach its threshold.

3.3542

risk threshold

1. measure of the level of uncertainty or the level of impact at which a stakeholder may have a specific interest. Below that risk threshold, the organization will accept the risk. Above that risk threshold, the organization will not tolerate the risk [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 2. condition that triggers some stakeholder action [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.16*]

Note 1 to entry: Different risk thresholds can be defined for each risk, risk category or combination of risks, based on differing risk criteria.

3.3543

risk tolerance

1. the degree, amount, or volume of risk that an organization or individual will withstand [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3544

risk transfer

1. transferring responsibility for managing a risk factor to another organization or functional entity better able to mitigate the risk factor

3.3545

risk transference

1. a risk response strategy whereby the project team shifts the impact of a threat to a third party, together with ownership of the response [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3546

risk treatment

1. process of selection and implementation of measures to modify risk [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.17*] 2. means of modifying risk 3. process to eliminate risk or reduce it to a tolerable level [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.20*]

Note 1 to entry: Risk treatment measures can include avoiding, optimizing, transferring, or retaining risk.

3.3547

risk trigger

1. predetermined threshold value of a risk metric that triggers invocation of a contingency plan when the risk metric crosses the threshold

3.3548

risk urgency assessment

1. review and determination of the timing of actions which may need to occur sooner than other risk items [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3549

risk-based testing

1. testing in which the management, selection, prioritization, and use of testing activities and resources are consciously based on corresponding types and levels of analyzed risk [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.35]

3.3550

risk-free MARR

1. MARR that has not been adjusted to address the risk in an alternative, because the risk is considered insignificant in the decision analysis

3.3551

RJE

1. remote job entry

3.3552

RM-ODP

1. Reference Model of Open Distributed Processing [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview; ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications*, 4] 2. Open Distributed Processing: Reference Model [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions*, 4]

3.3553

RMI

1. remote method invocation [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.3]

3.3554

robotics

1. techniques involved in designing, building, and using robots [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3555

robustness

1. degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions

cf. error tolerance, fault tolerance

3.3556

ROI

1. return on investment

3.3557

role

1. participation of an entity in a relationship [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2] 2. a defined function to be performed by a project team member, such as testing, filing, inspecting, coding [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] 3. expression of an object playing a part in a relationship [*ISO/IEC 15476-4:2005 Information technology — CDIF semantic metamodel — Part 4: Data models*, 6.5] 4. formal placeholder in the specification of a composite object that identifies those aspects of the behavior of some component object required for it to form part of the composite and links them as constraints on an actual object in an instance of the composite [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.17]

3.3558

role name

- 1.** name that more specifically names the nature of a related value class or state class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.167*] **2.** name assigned to a foreign key attribute to represent the use of the foreign key in the entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.167*]

Note 1 to entry: For a relationship, a role name is a name given to a class in a relationship to clarify the participation of that class in the relationship, that is, connote the role played by a related instance. For an attribute, a role name is a name used to clarify the sense of the value class in the context of the class for which it is a property.

3.3559

roll in

- 1.** to transfer data or computer program segments from auxiliary storage to main storage
cf. roll out, swap

3.3560

roll out

- 1.** to transfer data or computer program segments from main storage to auxiliary storage for the purpose of freeing main storage for other uses

cf. roll in, swap

3.3561

rolling wave planning

- 1.** an iterative planning technique in which the work to be accomplished in the near term is planned in detail, while the work in the future is planned at a higher level [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3562

ROM

- 1.** read-only memory

3.3563

root arrow segment

root

root segment

- 1.** arrow segment of a junction from which other arrow segments branch or to which other arrow segments join [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.99*]

3.3564

root cause

- 1.** source of a defect such that if it is removed, the defect is decreased or removed

3.3565

root cause analysis

- 1.** an analytical technique used to determine the basic underlying reason that causes a variance or a defect or a risk. A root cause may underlie more than one variance or defect or risk. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3566

root compiler

- 1.** compiler whose output is a machine independent, intermediate-level representation of a program

Note 1 to entry: A root compiler, when combined with a code generator, comprises a full compiler.

3.3567

root-cause analysis

- 1.** determination of a potential problem's (a risk factor's) underlying cause or causes.

3.3568

routine

1. subprogram that is called by other programs and subprograms **2.** function or procedure invocable for a single purpose **3.** program, or part of a program, that has some general or frequent use [*ISO/IEC 2382:2015, Information technology — Vocabulary*] *cf.* coroutine, subroutine

Note 1 to entry: The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages.

3.3569

RPC

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

3.3570

RTC

1. real-time clock

3.3571

RTM

1. requirements traceability matrix

3.3572

RTOS

1. real-time operating system

3.3573

rule

1. constraint on a system specification [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 11.2.7*] **2.** single column through the condition and action entry parts of the decision table, defining a unique set of conditions to be satisfied and the actions to be taken in consequence [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.4*]

Note 1 to entry: A rule is satisfied if all conditions meet the condition entries of the rule.

3.3574

Rule and Constraint Language (RCL)

1. declarative specification language that is used to express the realization of responsibilities and to state queries [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.168*] *cf.* specification language

3.3575

rule-based language

1. nonprocedural language that permits the user to state a set of rules and to express queries or problems that use these rules

cf. declarative language, interactive language

3.3576

run

1. in software engineering, a single, usually continuous, execution of a computer program **2.** to execute a computer program

cf. run time

3.3577

run time

running time

1. instant at which a computer program begins to execute **2.** period of time during which a computer program is executing

cf. execution time

3.3578

runtime platform

1. set of hardware and software components that implement the services utilized by the application software [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.3579

RUR

1. Reference User Requirements [*ISO/IEC TR 14143-4:2002 Information technology — Software measurement — Functional size measurement — Part 4: Reference model*, 4]

3.3580

RUSP

1. ready-to-use software product [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.6]

3.3581

RWD

1. responsive web design [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.3582

SAD

1. software architecture description [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.3583

safety

1. expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered

Note 1 to entry: See ISO/IEC Guide 51 for issues related to safety.

3.3584

safety criteria

1. limits of acceptable risk associated with a hazard [*ISO/IEC TS 15504-10:2011 Information technology — Process assessment — Part 10: Safety extension*, 3.4]

Note 1 to entry: These limits can be defined as imposed safety targets or developed from analysis or development policy.

3.3585

safety demonstration

1. body of evidence and rationale that shows an item is justified as being safe within allowed limits on risk [*ISO/IEC TS 15504-10:2011 Information technology — Process assessment — Part 10: Safety extension*, 3.3]

EXAMPLE: that an item was designed and integrated correctly to approved standards by competent people in accordance with approved procedures with sufficient mitigation, and tested sufficiently

3.3586

safety integrity requirement

1. likelihood of a safety-related system satisfactorily performing the required safety functions under stated conditions [*ISO/IEC TS 15504-10:2011 Information technology — Process assessment — Part 10: Safety extension*, 3.6]

3.3587

safety life cycle

1. project or product life cycle in which safety processes are performed [*ISO/IEC TS 15504-10:2011 Information technology — Process assessment — Part 10: Safety extension*, 3.7]

3.3588

safety requirement

1. requirement that is needed to ensure the safety of the product [ISO/IEC TS 15504-10:2011 *Information technology — Process assessment — Part 10: Safety extension*, 3.8]

3.3589

safety-critical software

1. software that falls into one or more of the following categories: a) software whose inadvertent response to stimuli, failure to respond when required, response out-of-sequence, or response in combination with other responses can result in an accident b) software that is intended to mitigate the result of an accident c) software that is intended to recover from the result of an accident [IEEE 1228-1994 (R2002) *IEEE Standard for Software Safety Plans*, 3.1.4]

3.3590

SAIV

1. schedule as independent variable [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3591

SAM

1. software asset management [ISO/IEC 19770-3:2016, *Information technology — IT asset management — Part 3: Entitlement schema*, 3.2]

3.3592

SAM owner

1. individual at a senior organization-wide level who is identified as being responsible for SAM [ISO/IEC 19770-1:2012 *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*, 3.11]

3.3593

SAM practitioner

1. individual involved in the practice or role of managing software assets [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.31]

Note 1 to entry: A SAM practitioner is often involved in the collection or reconciliation of software inventory and software entitlements.

3.3594

SAM program scope

1. clear statement listing of all parts of the organization and types of software, assets, and platforms covered by a SAM program [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.32]

3.3595

SAM tool

software asset management tool

1. software used to assist in and automate parts of the process of management of software assets [ISO/IEC 19770-3:2016, *Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.25]

3.3596

sample instance diagram

1. form of presenting example instances in which instances are shown as separate graphic objects [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.169]

cf. sample instance table

Note 1 to entry: The graphic presentation of instances can be useful when only a few instances are presented.

3.3597

sample instance table

1. form of presenting example instances in which instances are shown as a tabular presentation [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*,

3.1.170]

cf. sample instance diagram

Note 1 to entry: The tabular presentation of instances can be useful when several instances of one class are to be presented.

3.3598

SAR

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

cf. SRR

3.3599

SAS

1. security attribute service [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability, 5*]

3.3600

satisfaction

1. freedom from discomfort and positive attitudes towards the use of the product [*ISO/IEC 25062:2006 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports, 4.4*] **2.** user's subjective response when using the product [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation, 3.34*] **3.** degree to which user needs are satisfied when a product or system is used in a specified context of use [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.1.3*]

3.3601

satisficing

1. decision technique that discards any alternative with an attribute value outside an acceptable range
cf. dominance, lexicography

3.3602

SBI

1. serial bus interface

3.3603

SBS

1. system breakdown structure [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 4.2*]

3.3604

scaffolding

1. computer programs and data files built to support software development and testing, but not intended to be included in the final product
cf. programming support environment

EXAMPLE: dummy routines or files, test case generators, software monitors, stubs

3.3605

scalar

1. value that is atomic [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.171*]

cf. collection-valued

Note 1 to entry: That is, having no parts.

3.3606

scalar-valued class

- 1.** class in which each instance is a single value [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.173]
cf. collection-valued class

3.3607

scalar-valued property

scalar property

- 1.** property that maps to a scalar-valued class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.174]
cf. collection-valued property

3.3608

scale

- 1** ordered set of values, continuous or discrete, or a set of categories to which the attribute is mapped [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.34; *ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.30]

EXAMPLE: a nominal scale which corresponds to a set of categories, an ordinal scale which corresponds to an ordered set of scale points, an interval scale which corresponds to an ordered scale with equidistant scale points, and a ratio scale which not only has equidistant scale points but also possesses an absolute zero

Note 1 to entry: The type of scale depends on the nature of the relationship between values on the scale. Metrics using nominal or ordinal scales produce qualitative data, and metrics using interval and ratio scales produce quantitative data.

3.3609

scatter diagram

- 1.** a correlation chart that uses a regression line to explain or to predict how the change in an independent variable will change a dependent variable [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3610

SCCS

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

3.3611

scenario

script

- 1.** step-by-step description of a series of events that occur concurrently or sequentially

cf. use case

Note 1 to entry: A scenario can be a user story, use case, operational concept, or sequence of events the software may encounter.

3.3612

scenario testing

- 1.** class of test design technique in which tests are designed to execute individual scenarios [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.36]

3.3613

schedule as independent variable (SAIV)

- 1.** a date-certain scheduling method for a project with a specific end date, after which the value of the project declines precipitously or a penalty for non-completion is applied [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3614

schedule baseline

1. the approved version of a schedule model that can be changed only through formal change control procedures and is used as a basis for comparison to actual results [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3615

schedule compression

1. techniques used to shorten the schedule duration without reducing the project scope [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. crashing, fast tracking

3.3616

schedule data

1. the collection of information for describing and controlling the schedule [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3617

schedule development

1. process of creating the project schedule by analyzing activity sequences, activity durations, resource requirements, and schedule constraints

3.3618

schedule forecast

1. estimate or prediction of conditions and events in the project's future based on information and knowledge available at the time the schedule is calculated [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3619

schedule management plan

1. a component of the project management plan that establishes the criteria and the activities for developing, monitoring, and controlling the schedule [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3620

schedule model

1. a representation of the plan for executing the project's activities, including durations, dependencies and other planning information, used to produce a project schedule along with other scheduling artifacts [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3621

schedule network analysis

network analysis

schedule analysis

1. the technique of identifying early and late start dates, as well as early and late finish dates, for the uncompleted portions of project schedule activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. backward pass, critical path method, critical chain method, resource leveling

3.3622

schedule network templates

1. a set of activities and relationships that have been established that can be used repeatedly for a particular application area or an aspect of the project where a prescribed sequence is desired [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3623

schedule performance index (SPI)

1. a measure of schedule efficiency expressed as the ratio of earned value to planned value [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3624

schedule variance (SV)

- 1.** a measure of schedule performance expressed as the difference between the earned value and the planned value [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3625

scheduler

- 1.** computer program, usually part of an operating system, that schedules, initiates, and terminates jobs

3.3626

SCI

- 1.** Software Configuration Item [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management, 5.1*] **2.** serial communication interface

3.3627

SCM

- 1.** Software Configuration Management [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management, 5.1*]

3.3628

SCMP

- 1.** software configuration management plan

3.3629

SCMPI

- 1.** Software Configuration Management Planned Information.

3.3630

SCN

- 1.** specification change notice

3.3631

scope

- 1.** the sum of the products, services, and results to be provided as a project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** behavior that a system is expected to exhibit [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.1.1*]

cf. project scope, product scope, scope of the FSM

3.3632

scope change

- 1.** any change to the project scope. A scope change almost always requires an adjustment to the project cost or schedule [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3633

scope of the FSM

- 1.** the set of functional user requirements to be included in a specific FSM instance [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.11; ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, A.12*] **2.** set of functional user requirements to be included in a specific functional size measurement instance [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.24*]

EXAMPLE: If an organization needs to know the size of its software portfolio, then the Scope of the FSM will include all the functional user requirements currently utilized. However, if a project manager is seeking to determine the size of a particular release of software, then the scope will include only those functional user requirements impacted by the project.

Note 1 to entry: The Scope of the FSM is determined by the purpose for measuring the software.

3.3634

screen capture

screen dump

1. representation of what the user will see while using the software

3.3635

scripted testing

1. dynamic testing in which the tester's actions are prescribed by written instructions in a test case [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.37] 2. testing performed based on a documented test script [ISO/IEC/IEEE 29119-2:2013 *Software and systems engineering — Software testing — Part 2: Test processes*, 4.23]

Note 1 to entry: normally applies to manually executed testing, rather than the execution of an automated script

3.3636

SCRM

1. supply chain risk management [IEEE 15288.2:2014 *IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.3637

scrum

1. iterative project management framework used in agile development, in which a team agrees on development items from a requirements backlog and produces them within a short duration of a few weeks [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.9]

3.3638

scrum master

1. person who facilitates the scrum process within a team or project [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.10]

3.3639

scrum meeting

1. brief daily project status meeting or other planning meeting in agile development methodologies [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.11]

Note 1 to entry: The scrum meeting is usually chaired by the scrum master.

3.3640

scrum report

1. report that documents the daily activities of a scrum team, recording any problems or issues to be dealt with [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.12]

3.3641

scrum team

1. members of an agile development team working together under the scrum process, usually led by the scrum master and project owner [ISO/IEC/IEEE 26515: 2011 *Systems and software engineering: Developing user documentation in an agile environment*, 4.13]

3.3642

SDa

1. submit data activity [ISO/IEC 29155-2:2013: *Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*, 4]

3.3643

SDD

1. software design description [IEEE 1012-2012 *IEEE Standard for System and Software Verification and Validation*] 2. software design document 3. system design document [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]

3.3644

SDP

1. software development plan [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*]

3.3645

SDR

1. system design review

3.3646

SDRAM

1. synchronous dynamic random access memory

3.3647

SE

1. Software Engineering [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*]

3.3648

SECIOP

1. Secure Inter-ORB Protocol [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, Secure Inter-ORB Protocol*]

3.3649

second normal form

1. result of a normalization process that transforms groups of data so that each non-key attribute depends on the key attribute(s) of the group of data and all parts of the key attribute(s)

3.3650

secondary risk

1. a risk that arises as a direct result of implementing a risk response [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3651

secondary user

1. person who interacts with the product to support the primary users
cf. operator

EXAMPLE: content provider, system manager, administrator, security manager, maintainer, installer

3.3652

secondary window

1. window that contains information that depends on information in another window (the primary window) [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.43*]

3.3653

security

1. protection against intentional subversion or forced failure [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.41*] **2.** defining, achieving, and maintaining confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of a system **3.** degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, 4.2.6*] **4.** protection of computer hardware or software from accidental or malicious access, use, modification, destruction, or disclosure [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1*] **5.** protection of information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.39*] **6.** protection against intentional subversion or forced failure, containing a composite of four attributes: confidentiality, integrity, availability and accountability, plus

aspects of a fifth, usability, all of which have the related issue of their assurance [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.45]

Note 1 to entry: A composite of four attributes: confidentiality, integrity, availability, and accountability, plus aspects of a fifth: usability, all of which have the related issue of their assurance. Security pertains to personnel, data, communications, and the physical protection of computer installations.

3.3654

security accreditation

1. formal declaration by management that an IT system is approved to operate in a particular security mode using a prescribed set of safeguards at an acceptable level of risk **2.** independent accreditation body's certification that an IT system meets a predetermined security standard.

3.3655

security authority

1. administrator responsible for the implementation of a security policy [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 15.1.2]

3.3656

security branch

1. branch, created at the time of a release, to which only security commits are made

3.3657

security domain

1. domain in which the members are obliged to follow a security policy established and administered by a security authority [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 15.1.3]

Note 1 to entry: The security authority is the controlling object for the security domain.

3.3658

security interaction policy

1. those aspects of the security policies of different security domains that are necessary in order for interactions to take place between those domains [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 15.1.4]

3.3659

security kernel

1. small, self-contained collection of key security-related statements that works as a privileged part of an operating system, specifying and enforcing criteria that must be met for programs and data to be accessed

3.3660

security policy

1. rules for need-to-know and access-to-information at each project organization level **2.** set of rules that constrains one or more sets of activities of one or more sets of objects [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 15.1.1]

3.3661

security testing

1. type of testing conducted to evaluate the degree to which a test item, and associated data and information, are protected so that unauthorized persons or systems cannot use, read, or modify them, and authorized persons or systems are not denied access to them [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.38]

3.3662

SEE

1. Software Engineering Environment [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 5]

3.3663

SEE service

- 1.** one or more service operations to support life cycle activities for the SEE [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.8*]

Note 1 to entry: A SEE Service supplier provides a SEE Service for a SEE Service acquirer.

3.3664

SEE service acquirer

- 1.** actor that acquires an SEE service [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.12*]

3.3665

SEE service supplier

- 1.** actor that supplies an SEE service [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.13*]

3.3666

segment

- 1.** one of the subsystems or combinations of subsystems that make up an overall system **2.** in storage allocation, a self-contained portion of a computer program that can be executed without maintaining the entire program in main storage **3.** collection of data that is stored or transferred as a unit **4.** in path analysis, a sequence of computer program statements between two consecutive branch points **5.** to divide a system, computer program, or data file into segments as in (1), (2), or (3) **6.** collection of data that corresponds to one or more coherent views of a system of interest that is stored or transferred as a unit [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

EXAMPLE: the accounts payable segment of a financial system

3.3667

segmented executor

- 1.** set of physically distinct artifacts, a physical partition of the executor [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

Note 1 to entry: Each segment encapsulates independent state and is capable of being independently activated. Each segment provides at least one facet.

3.3668

SEI

- 1.** serial expansion interface **2.** Software Engineering Institute

3.3669

selected sellers

- 1.** the sellers which have been selected to provide a contracted set of services or products [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3670

selective dump

- 1.** dump of designated storage location areas only

cf. change dump, dynamic dump, memory dump, postmortem dump, snapshot dump, static dump

3.3671

selective trace

- 1.** variable trace that involves only selected variables

cf. execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

3.3672

self-contained

1. process in which no prior or subsequent processing steps are needed to initiate or complete the functional user requirements [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.47*]

3.3673

self-descriptiveness

1. degree to which a system or component contains enough information to explain its objectives and properties **2.** software attributes that explain a function's implementation
cf. maintainability, testability, usability

3.3674

self-documented

1. pertaining to source code that contains comments explaining its objectives, operation, and other information useful in understanding and maintaining the code

3.3675

self-relative address

1. address that must be added to the address of the instruction in which it appears to obtain the address of the storage location to be accessed

cf. base address, indexed address, offset, relative address

3.3676

seller

1. a provider or supplier of products, services, or results to an organization [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. supplier

3.3677

seller proposals

1. formal responses from sellers to a request for proposal or other procurement document specifying the price, commercial terms of sale, and technical specifications or capabilities the seller will do for the requesting organization that, if accepted, would bind the seller to perform the resulting agreement [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3678

semantic agreement

1. passive interconnection in which two things agree on a common interpretation of statements (symbol arrangements) by reference to a shared thing or phenomenon [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.12*]

cf. syntactic agreement

3.3679

semantic error

1. error resulting from a misunderstanding of the relationship of symbols or groups of symbols to their meanings in a given language

cf. syntactic error

3.3680

Semantic Transfer Language (STL)

1. language for the purpose of representing software application behavior descriptions [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

Note 1 to entry: The Semantic Transfer Language has a formal syntax that is computer-parsable, while remaining easy for users to read and write.

3.3681

Semantic Transfer Language (STL) clause

1. portion of an STL sentence that describes and characterizes an attribute of, or a relationship for, a software behavior concept [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.3682

Semantic Transfer Language (STL) concept

1. software behavior concept that is represented in the STL [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.3683

Semantic Transfer Language (STL) sentence

1. statement conforming to the syntax of the Semantic Transfer Language (STL) that describes and characterizes a software behavior concept recognized within the STL [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.3684

semantics

1. meaning of the syntactic components of a language [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.175*] **2.** relationships of symbols or groups of symbols to their meanings in a given language

cf. syntax

3.3685

semaphore

1. shared variable used to synchronize concurrent processes by indicating whether an action has been completed or an event has occurred

cf. flag, indicator

3.3686

SEMDM

1. software engineering metamodel for development methodologies [*ISO/IEC 24744:2014 Software Engineering — Metamodel for development methodologies, 4.2*]

3.3687

semiconductor

1. substance with conductive properties between those of a conductor and an insulator

EXAMPLE: silicon or germanium used in electronic circuits

3.3688

SEMP

1. Systems Engineering Management Plan [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 3*]

3.3689

sensitivity analysis

1. a quantitative risk analysis and modeling technique used to help determine which risks have the most potential impact on the project. It examines the extent to which the uncertainty of each project element affects the objective being examined when all other uncertain elements are held at their baseline values. The typical display of results is in the form of a tornado diagram [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** risk-analysis technique that studies how changes in the values of estimated parameters affects the desirability of an alternative. Parameters where small changes in estimated values cause larger changes in desirability are said to be more sensitive. Sensitivity analysis guides the decision maker in identifying the estimated parameters (the sensitive ones) that deserve more careful study to improve the accuracy of the estimate.

3.3690

sentence

- 1.** linguistic construct containing one or more terms and predicates [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 7.2*]

Note 1 to entry: A sentence can express a proposition about the entities to which the terms refer. A predicate in a sentence can refer to a relationship between the entities referred to by the terms it links.

3.3691

SEP

- 1.** Systems Engineering Plan [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 3.5*]

3.3692

separate documentation

- 1.** documentation that can be used independently of the software [*ISO/IEC/IEEE 26512:2011 Systems and software engineering — Requirements for acquirers and suppliers of user documentation, 4.28*]
cf. embedded documentation

EXAMPLE: printed manual, standalone hypertext system

3.3693

sequence activities

- 1.** the process of identifying and documenting relationships among the project activities [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3694

sequence diagram

- 1.** a Unified Modeling Language (UML) diagram that depicts time-sequential ordering of interactions, as in a use case scenario of interactions between an actor and some system elements. Can be used to depict sequential and concurrent data flow or process flow [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3695

sequential

- 1.** pertaining to the occurrence of two or more events or activities in such a manner that one must finish before the next begins
cf. consecutive, serial

3.3696

sequential clustering

- 1.** task-structuring criterion in which objects that are constrained to execute sequentially are mapped to a task

3.3697

sequential cohesion

- 1.** type of cohesion in which the output of one task performed by a software module serves as input to another task performed by the module
cf. coincidental cohesion, communicational cohesion, functional cohesion, logical cohesion, procedural cohesion, temporal cohesion

3.3698

serial

- 1.** pertaining to the sequential transfer, occurrence, or processing of the individual parts of a whole, such as the bits of a character, using the same facilities for successive parts
cf. parallel (1), sequential

3.3699

serial bus

- 1.** shared channel that transmits data sequentially, bit-by-bit

EXAMPLE: IEEE 802 Ethernet uses a serial bus.

3.3700

serial bus interface (SBI)

1. connection for bidirectional serial data communication

EXAMPLE: RS232, RS485

Note 1 to entry: based on the IEEE 1394 standards

3.3701

serial communication interface (SCI)

1. unit that enables the serial exchange of data between a microprocessor and peripherals

3.3702

serial construct

sequential construct

1. program construct consisting of a sequence of steps not involving a decision or loop

3.3703

serial expansion interface (SEI)

1. connection to serve multiple serial channels

3.3704

serial peripheral interface (SPI)

1. synchronous (full duplex) serial communication interface used for two devices in embedded systems

3.3705

server

1. hardware system or software program which provides a service to clients 2. process implementing one or more operations on one or more objects [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.14]

3.3706

server object

1. object which performs some service on behalf of a client object [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.4.6]

3.3707

server-side

1. node, cluster or capsule, which: a) contains, or is potentially capable of containing, a basic engineering object that corresponds to a computational server object and stub, binder and protocol objects in a channel supporting operations involving the server object; or b) contains, or is a potentially capable of containing, a protocol object which (possibly via interactions with other engineering objects) can return a reply identifying another server-side [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions*, 3.3.10]

3.3708

service

1. means of delivering value for the customer by facilitating results the customer wants to achieve 2. performance of activities, work, or duties [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.35; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.46; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.42] 3. behavior, triggered by an interaction, which adds value for the service users by creating, modifying, or consuming information; the changes become visible in the service provider's environment [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.3.1] cf. product, result, deliverable

Note 1 to entry: A service is generally an intangible product. A service can be composed of other services. A service is self-contained, coherent, discrete.

3.3709

service alternative

1. alternative that is assumed to provide equivalent service to another alternative over their lives; all the revenue cash flows are ignored to simplify the comparison and only the expense cash flows are shown
cf. revenue alternative

3.3710

service component

1. single unit of a service that when combined with other units will deliver a complete service **2.** CORBA component with behavior, no state, and no identity [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.1*]

EXAMPLE: hardware, software, tools, applications, documentation, information, processes, or supporting services

Note 1 to entry: A service component can consist of one or more configuration items.

3.3711

service continuity

1. capability to deliver a service without interruption, or with consistent availability as scheduled and agreed

3.3712

service delivery profile

1. profile targeted at very small enterprises (VSEs) that need to perform and manage service delivery processes, either for systems or software products that they have developed or that were developed by others [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 3.54*]

3.3713

service desk

1. customer-facing support group for centralized resolution of incidents, change requests, and complaints concerning a service

3.3714

service export

1. interaction with the trading function in which a service offer is advertised, by adding the service offer to an identified set of service offers [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 14.5.1.2*]

3.3715

service import

1. interaction with the trading function which searches an identified set of service offers to discover interfaces at which a service satisfying a specified type is available [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 14.5.1.3*]

3.3716

service level agreement (SLA)

1. documented agreement between the service provider and customer that identifies services and service targets

Note 1 to entry: A service level agreement can also be established between the service provider and a supplier, an internal group, or a customer acting as a supplier. A service level agreement can be included in a contract or another type of documented agreement. An Application Management organization can be the service provider, but it can also be a customer itself, of another supplier.

3.3717

service management

1. set of capabilities and processes to direct and control the service provider's activities and resources for the design, transition, delivery and improvement of services to fulfill the service requirements

3.3718

service management system (SMS)

1. management system to direct and control the provision of one or more services

Note 1 to entry: The SMS includes all service management policies, objectives, plans, processes, documentation and resources required for the design, transition, delivery and improvement of services and to fulfill the requirements.

3.3719

service offer

- 1.** information describing an interface, how to bind to it, and the service that can be invoked using it [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 14.5.1.1]

3.3720

service primitive

- 1.** abstract definition of an interaction of channel objects that causes protocol exchanges between the protocol objects in the channel [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions*, 3.3.11]

3.3721

service provider

- 1.** organization that manages and delivers a service or services to the customer

Note 1 to entry: A customer can be internal or external to the service provider's organization.

3.3722

service request

- 1.** request for information, advice, access to a service, or a pre-approved change [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.25]

EXAMPLE: a request to provide access to a controlled application, a request to move hardware

3.3723

session component

- 1.** CORBA component with behavior, transient state, and identity that is not persistent [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

3.3724

set

- 1.** collection class with no duplicate members and where order is irrelevant [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.176] cf. bag, list

3.3725

set-up time

- 1.** period of time during which a system or component is being prepared for a specific operation cf. busy time, down time, idle time, setup time, up time

3.3726

SETA

- 1.** systems engineering and technical assistance [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.3727

SETR

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

3.3728

seven basic quality tools

- 1.** a standard toolkit used by quality management professionals who are responsible for planning, monitoring, and controlling the issues related to quality in an organization [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

EXAMPLE: cause-and-effect diagrams, checksheets, control charts, flowcharts, histograms, Pareto diagrams, scatter diagrams

3.3729

SF

- 1.** start-to-finish [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3730

SFR

- 1.** system functional review [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*] **2.** special function register

3.3731

SGML

- 1.** Standard Generalized Markup Language [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.3732

SGRAM

- 1.** synchronous graphics random access memory

3.3733

SHA

- 1.** secure hash algorithm [*ISO/IEC 19770-2:2015 Information technology — Software asset management — Part 2: Software identification tag, 3.2*]

3.3734

shadow class

- 1.** class presented in a view that is specified in some other view [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.177*]

3.3735

shell

- 1.** computer program or routine that provides an interface between the user and a computer system or program

3.3736

should-cost estimate

- 1.** estimate of the cost of a product or service used to evaluate the reasonableness of a supplier's proposed price

3.3737

shrink small outline package (SSOP)

- 1.** thinner rectangular surface mount integrated circuit unit with gull-wing leads on the two long sides

3.3738

signal

- 1.** atomic shared action resulting in one-way communication from an initiating object to a responding object [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.1*] **2.** variation of a physical quantity used to represent data [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: A signal is an interaction.

3.3739

signal interface

- 1.** interface in which all the interactions are signals [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.6*]

3.3740

signal interface signature

- 1.** interface signature for a signal interface [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.11*]

Note 1 to entry: A signal interface signature comprises a finite set of action templates, one for each signal type in the interface. Each action template comprises the name for the signal, the number, names and types of its parameters, and an indication of causality (initiating or responding, but not both) with respect to the object which instantiates the template.

**3.3741
signature**

1. definition of the parameters of a given operation, including their number order, data types, and passing mode; the results if any; and the possible outcomes (normal vs. exceptional) that might occur [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.15] **2.** mathematical structure comprising a set of sorts and a set of operators [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.22] **3.** statement of what the interface to a responsibility "looks like". [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.178]

Note 1 to entry: A signature consists of the responsibility name, along with a property operator and the number and type of its arguments, if any. A type (class) can be specified for each argument to limit the argument values to being instances of that class.

**3.3742
signpost**

1. text, symbol, or small graphic that helps the user identify where particular types of information are located or where the information in the current display fits into the whole document [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.45]

Note 1 to entry: Information of different types can be indicated by symbols or graphics of different types.

3.3743

SIL

1. system integration laboratory [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.3744

SIM

1. [mobile telecommunications] subscriber identity module **2.** [embedded software] system integration module

3.3745

simple buffering

1. buffering technique in which a buffer is allocated to a computer program for the duration of the program's execution

cf. dynamic buffering

3.3746

simple token

1. valueless token, normally represented by a black dot, and used in place/transition nets (as opposed to high-level nets) [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.25.2]

3.3747

simplex receptacle

1. specialization of a receptacle that only allows a single connection at a given time [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*, 4.1]

3.3748

simplicity

1. degree to which a system or component has a design and implementation that is straightforward and easy to understand **2.** software attributes that provide implementation of functions in the most understandable manner *cf.* complexity

3.3749

simulation

1. model that behaves or operates like a given system when provided a set of controlled inputs **2.** process of developing or using a model as in (1) **3.** use of a data processing system to represent selected behavioral characteristics of a physical or abstract system [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **4.** A simulation uses a project model that translates the uncertainties specified at a detailed level into their potential impact on objectives that are expressed at the level of the total project. Project simulations use computer models and estimates of risk, usually expressed as a probability distribution of possible costs or durations at a detailed work level, and are typically performed using Monte Carlo analysis [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. emulation

3.3750

simulator

1. device, computer program, or system that behaves or operates like a given system when provided a set of controlled inputs

cf. emulator

3.3751

simultaneous

1. pertaining to the occurrence of two or more events at the same instant of time

cf. concurrent

3.3752

simultaneous recursion

1. situation in which two software modules call each other

3.3753

single boot

1. having only one boot mode to start a computer

cf. dual boot

3.3754

single-hit decision table

1. decision table where any set of conditions will be satisfied by one, and only one, rule [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.2*]

3.3755

single-level encoding

1. microprogramming technique in which different microoperations are encoded as different values in the same field of a microinstruction

cf. two-level encoding

3.3756

single-step operation

single-step execution

step-by-step operation

1. debugging technique in which a single computer instruction, or part of an instruction, is executed in response to an external signal

3.3757

single-valued property

1. property with a single-valued mapping [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.179*]

cf. multi-valued property

3.3758

SIP

1. system integration plan [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.3759

site map

- 1.** textual or graphical overview of the navigation structure of a website [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.23]

3.3760

sizing

- 1.** process of estimating the amount of computer storage or the number of source lines required for a software system or component

cf. timing

3.3761

skeleton

- 1.** object-interface-specific ORB component which assists an object adapter in passing requests to particular methods [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.16]

3.3762

sku

- 1.** stock keeping unit [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.48]

3.3763

SLA

- 1.** service level agreement

3.3764

SLC

- 1.** software life cycle [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.3]

3.3765

SLCP

- 1.** Software Life Cycle Processes

3.3766

SLOC

- 1.** Source Lines of Code, the number of lines of programming language code in a program before compilation [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual*, 10]

3.3767

slush

- 1.** preparation for a feature or code freeze

Note 1 to entry: During this period, developers will commit code they have been working on but are discouraged from starting on new elements. If a freeze lasts for a long time, a slush might be introduced to ease its passing by allowing in some extra elements.

3.3768

Small and Medium Enterprise (SME)

- 1.** enterprises which employ fewer than 250 persons and which have an annual turnover not exceeding 50 million euro, and/or an annual balance sheet total not exceeding 43 million euro [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile*, 3.6] **2.** enterprise which employs fewer than 250 persons [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.52]

3.3769

small outline integrated circuit (SOIC)

- 1.** surface-mounted integrated circuit (chip) package with gull-wing pins extending outward

3.3770

small outline package (SOP)

- 1.** rectangular surface mount integrated circuit unit with gull-wing leads on the two long sides

3.3771

SMART

- 1.** specific, measurable, achievable, relevant and traceable [*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*]

3.3772

SME

- 1.** subject-matter expert [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*] **2.** small and medium enterprise [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 3.6*]

3.3773

SMIR

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

3.3774

SMS

- 1.** service management system

3.3775

SMT

- 1.** self-managed transaction [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, Secure Inter-ORB Protocol*]

3.3776

SN

- 1.** symmetric net [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.2.7*]

3.3777

snapshot dump

- 1.** dynamic dump of the contents of one or more specified storage areas

cf. change dump, dynamic dump, memory dump, postmortem dump, selective dump, static dump

3.3778

SNCS

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

3.3779

snoop

- 1.** technique for monitoring bus performance in multi-layer memory cache

EXAMPLE: bus snoop, cache snoop, bus sniffing

3.3780

SoC

- 1.** system on a chip

3.3781

soft copy image

- 1.** nonpermanent output of information in audio or visual format [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: a video display

3.3782

soft failure

- 1.** failure that permits continued operation of a system with partial operational capability
cf. hard failure

3.3783

software

SW

- 1.** computer programs, procedures and possibly associated documentation and data pertaining to the operation of a computer system [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*] **2.** all or part of the programs, procedures, rules, and associated documentation of an information processing system [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1; ISO/IEC 19770-1:2012 Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance, 3.14*] **3.** program or set of programs used to run a computer [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.46*]
cf. application software

EXAMPLE: command files, job control language, both executable and non-executable software, such as fonts, graphics, audio and video recordings, templates, dictionaries, and documents

Note 1 to entry: include both executable and non-executable software, such as fonts, graphics, audio and video recordings, templates, dictionaries, documents, and information structures such as database records

3.3784

software acquisition process

- 1.** period of time that begins with the decision to acquire a software product and ends when the product is no longer available for use.

3.3785

software artifact

- 1.** tangible machine-readable document created during software development [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.3786

software asset

- 1.** description of a partial solution (such as a component or design document) or knowledge (such as requirements database or test procedures) that engineers use to build or modify software products [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

3.3787

software asset management (SAM)

- 1.** control and protection of software and related assets within an organization, and control and protection of information about related assets which are needed in order to control and protect software assets. [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.35*]

Note 1 to entry: infrastructure and processes necessary for the effective management, control and protection of the software assets within an organization, throughout all stages of their lifecycle

3.3788

software baseline

- 1.** set (one or more) of software configuration items formally designated and fixed at a specific time during the software life cycle [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]

3.3789

software behavior concepts

- 1.** types of quantifiable properties and relationships whose instances describe the dynamic behavior of software execution [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.3790

software characteristic

- 1.** inherent, possibly accidental, trait, quality, or property of software.

EXAMPLE: functionality, performance, attributes, design constraints, number of states, lines of branches

3.3791

software component (SC)

- 1.** entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.36] **2.** functionally or logically distinct part of a software configuration item, distinguished for the purpose of convenience in designing and specifying a complex SCI as an assembly of subordinate elements **3.** software system or element [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.32]

EXAMPLE: data, document, module, unit

Note 1 to entry: Software component refers to a part of a whole, such as a component of a software product or a component of a software identification tag.

3.3792

software configuration item (SCI)

- 1.** software entity that has been established as a configuration item

cf. computer software component, computer software configuration item, hardware configuration item, software item

Note 1 to entry: The SCI exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the software has been established as a configurable item.

3.3793

software configuration management (SCM)

- 1.** process of applying configuration management throughout the software life cycle to ensure the completeness and correctness of SCIs [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management*, 4.4]

3.3794

software consumer

- 1.** entity that uses an entitlement of a software package [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.37]

3.3795

software creator

- 1.** person or organization that creates a software product or package [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.38]

Note 1 to entry: This entity might or might not own the rights to sell or distribute the software.

3.3796

software design

- 1.** use of scientific principles, technical information, and imagination in the definition of a software system to perform pre-specified functions with maximum economy and efficiency

3.3797

software design audit

- 1.** review of a software product to determine compliance with requirements, standards, and contractual documents

3.3798

software design concept

- 1.** fundamental idea (such as information hiding) that can be applied to designing a system

3.3799

software design description (SDD)

- 1.** representation of software created to facilitate analysis, planning, implementation, and decision-making [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.28*]

Note 1 to entry: The software design description is used as a medium for communicating software design information and can be thought of as a blueprint or model of the system.

3.3800

software design notation

software design representation

- 1.** means of describing a software design

EXAMPLE: structure charts and pseudocode

Note 1 to entry: It can be diagrammatic, symbolic, or textual.

3.3801

software design verification

- 1.** evaluation of a design to determine correctness with respect to stated requirements, conformance to design standards, system efficiency, and other criteria

3.3802

software developer

- 1.** person who creates software

Note 1 to entry: Often a software developer works with other developers for a software manufacturer to create commercial applications. A software developer can also often work as an in-house developer of software for use by the software developer's own organization.

3.3803

software development cycle

- 1.** period of time that begins with the decision to develop a software product and ends when the software is delivered

cf. software life cycle

Note 1 to entry: This cycle typically includes a requirements phase, design phase, implementation phase, test phase, and sometimes, installation and checkout phase. The phases listed above can overlap or be performed iteratively, depending upon the software development approach used. This term is sometimes used to mean a longer period of time, either the period that ends when the software is no longer being enhanced by the developer, or the entire software life cycle.

3.3804

software development file (SDF)

software development folder

software development notebook

unit development folder

- 1.** collection of material pertinent to the development of a given software unit or set of related units

Note 1 to entry: Contents typically include the requirements, design, technical reports, code listings, test plans, test results, problem reports, schedules, and notes for the units.

3.3805

software development library

project library

program support library

1. software library containing computer readable and human readable information relevant to a software development effort

cf. master library, production library, software repository, system library

3.3806

software development plan (SDP)

1. project plan for a software development project

3.3807

software development process

1. process by which user needs are translated into a software product

Note 1 to entry: The process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes, installing and checking out the software for operational use. These activities can overlap or be performed iteratively.

3.3808

software diversity

1. software development technique in which two or more functionally identical variants of a program are developed from the same specification by different programmers or programming teams with the intent of providing error detection, increased reliability, additional documentation, or reduced probability that programming or compiler errors will influence the end results

3.3809

software element

1. system element that is software

cf. system element, software/system element

3.3810

software engineering

SE

SWE

1. systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **2.** application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software [ISO/IEC TR 19759:2016, *Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)*]

3.3811

software engineering environment (SEE)

1. environment that provides automated system context services and software-specific services for the engineering of software systems and related domains, such as project management and process management [ISO/IEC 15940:2013 *Systems and software engineering — Software Engineering Environment Services*, 2.7] **2.** hardware, software, and firmware used to perform a software engineering effort [IEEE 730-2014 *IEEE Standard for Software Quality Assurance Processes*, 3.2]
cf. infrastructure

Note 1 to entry: It includes the platform, system software, utilities, and CASE tools installed.

3.3812

software entitlement

1. software license use rights as defined through agreements between a software licensor and a software consumer [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.39]

Note 1 to entry: Effective use rights take into account any contracts and all applicable licenses, including full licenses, upgrade licenses, and maintenance agreements.

3.3813

software entitlement reconciliation

1. process of comparing software entitlements owned with those required (granted and deployed), usually to determine compliance with software license agreements [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.30*]

3.3814

software feature

1. software characteristic specified or implied by requirements documentation.

EXAMPLE: functionality, performance, attributes, or design constraints

3.3815

software function

1. implementation of an algorithm in the software with which the end user or the software can perform part or all of a work task [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4.1.14*]

Note 1 to entry: a function does not need to be callable by the end user (e.g., automatic backup or data saving).

3.3816

software hazard

1. software condition that is a prerequisite to an accident [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.5*] cf. system hazard

3.3817

software identification tag

SWID tag

SWID

1. information structure containing identification information about a software configuration item, which can be authoritative if provided by a software creator [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.40*] **3.** set of structured data elements containing authoritative identification information about a software configuration item [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.31*]

3.3818

software item

1. aggregation of software, such as a computer program or database, that satisfies an end use function and is designated for specification, qualification testing, interfacing, configuration management, or other purposes [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*] **2.** source code, object code, control code, control data, or a collection of these items [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.41; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.47*] **3.** identifiable part of a software product [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1; ISO/IEC 90003:2014 Software engineering — Guidelines for the application of ISO 9001:2008 to computer software, 3.14*] cf. computer software component, computer software configuration item, software configuration item

EXAMPLE: identification and descriptions of the software product, software life-cycle data, archive and release data, and instructions for building the executable object code

3.3819

software library

program library

1. controlled collection of software and related documentation designed to aid in software development, use, or maintenance **2.** controlled collection of SCIs to aid in development, operation and maintenance [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management, 4.5*]

Note 1 to entry: Types include master library, production library, software development library, software repository, system library.

3.3820

software license

1. legal rights to use software in accordance with terms and conditions specified by the software licensor [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.41]

Note 1 to entry: Using a software product can include accessing, copying, distributing, installing and executing the software product, depending on the license's terms and conditions

3.3821

software licensee

1. person or organization granted a license to use a specific software product [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.33]

3.3822

software licensor

1. person or organization who owns or holds the rights to issue a software license for a specific software package [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.43]

3.3823

software life cycle (SLC)

1. project-specific sequence of activities that is created by mapping the activities of a standard onto a selected software life cycle model (SLCM) [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2] **2.** software system or software product cycle initiated by a user need or a perceived customer need and terminated by discontinued use of the product or when the software is no longer available for use

3.3824

software maintenance

1. totality of activities required to provide cost-effective support to a software system [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance*, 3.1] **3.** entitlement of additional rights (such as additional functionality, upgrade or support) for a previously granted software entitlement [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.35]

Note 1 to entry: Pre-delivery activities include planning for post-delivery operations, supportability, and logistics determination. Post-delivery activities include software modification, training, and operating a help desk.

3.3825

software monitor

1. software tool that executes concurrently with another program and provides detailed information about the execution of the other program
cf. hardware

3.3826

software package

1. complete and documented set of software supplied for a specific application or function [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.44]
cf. software product

Note 1 to entry: the set of files associated with a specific set of business functionalities that can be installed on a computing device and has a set of specific licensing requirements

3.3827

software packager

1. entity that packages or bundles software created by others [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.45]

EXAMPLE a value -added reseller who bundles a software package to work with an embedded system, or a software reseller who is licensed to combine a number of different software products into a single bundle

3.3828

software piracy

1. illegal use or copying of software products [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3829

software product

1. set of computer programs, procedures, and possibly associated documentation and data [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.42; *ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE* 4.31; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.48] 2. any of the individual items in (1) 3. complete set of software designed for delivery to a software consumer or end-user, which can include computer programs, procedures and associated documentation and data. [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.46] 4. set of computer programs, procedures, database- and other data structure descriptions and associated documentation [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management*, 4.33]
cf. software package

Note 1 to entry: A software product can be designated for delivery, an integral part of another product, or used in development. Software products vary from large customized application software for one customer to standard software packages that are sold off the shelf to millions of customers.

3.3830

software product developer

1. person or organization that manufactures a software product [*ISO/IEC 14598-5:1998 Information technology — Software product evaluation — Part 5: Process for evaluators*, 4.7]

3.3831

software product evaluation

1. technical operation that consists of producing an assessment of one or more characteristics of a software product according to a specified procedure [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.32]

3.3832

software project life cycle (SPLC)

1. portion of the entire software life cycle applicable to a specific project.

Note 1 to entry: It is the sequence of activities created by mapping the activities of IEEE Std 1074 onto a selected software project life cycle model (SPLCM).

3.3833

software project life cycle model (SPLCM)

1. framework selected by each using organization on which to map the activities of IEEE Std 1074 to produce the software project life cycle (SPLC).

3.3834

software project life cycle process (SPLCP)

1. project-specific description of the process developed by adding the organizational process assets (OPAs) to the software project life cycle (SPLC) and the OPAs.

3.3835

software quality

1. capability of software product to satisfy stated and implied needs when used under specified conditions [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.33] 2. degree to which a software product satisfies stated and implied needs when used under specified conditions [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.13] 3. degree to which a software product meets established requirements [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2]

Note 1 to entry: Quality depends upon the degree to which the established requirements accurately represent stakeholder needs, wants, and expectations. In SQuaRE standards software quality has the same meaning as software product quality.

3.3836

software quality assurance (SQA)

1. a set of activities that assess adherence to, and the adequacy of the software processes used to develop and modify software products. SQA also determines the degree to which the desired results from software quality control are being obtained [*Software Extension to the PMBOK® Guide Fifth Edition*] 2. set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2]

Note 1 to entry: A key attribute of SQA is the objectivity of the SQA function with respect to the project. The SQA function can also be organizationally independent of the project; that is, free from technical, managerial, and financial pressures from the project.

3.3837

software quality characteristic

1. category of software quality attributes that bears on software quality [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.34]

Note 1 to entry: Software quality characteristics can be refined into multiple levels of sub-characteristics and finally into software quality attributes.

3.3838

software quality control (SQC)

1. a set of activities that measure, evaluate and report on the quality of software project artifacts throughout the project life cycle [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3839

software quality evaluation

1. systematic examination of the extent to which a software product is capable of satisfying stated and implied needs [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.35]

3.3840

software quality management

1. coordinated activities to direct and control an organization with regard to software quality [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2]

3.3841

software quality metric

1. function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.24]

3.3842

software quality requirement

1. requirement that a software quality attribute be present in software [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.14]

3.3843

software release management

1. management of the activities surrounding the release of one or more versions of software to one or more customers, including identifying, packaging, and delivering the elements of a product [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1]
cf. software configuration management, version

3.3844

software reliability

1. probability that software will not cause the failure of a system for a specified time under specified conditions

Note 1 to entry: The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered.

3.3845

software reliability management

1. process of optimizing the reliability of software through a program that emphasizes software error prevention, fault detection and removal, and the use of measurements to maximize reliability in light of project constraints such as resources (cost), schedule, and performance

3.3846

software repository

1. software library providing permanent, archival storage for software and related documentation [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]

cf. master library, production library, software development library, system library

3.3847

software requirement

1. software capability needed by a user to solve a problem or to achieve an objective **2.** software capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document

3.3848

software requirements analysis

1. process of studying user needs to arrive at a definition of system, hardware, or software requirements

3.3849

software requirements engineering

1. science and discipline concerned with analyzing and documenting software requirements **2.** software requirements elicitation, analysis, specification, verification, and management

cf. requirements engineering

Note 1 to entry: It involves transforming system requirements into a description of software requirements, performance parameters, and a software configuration using an iterative process of definition, analysis, trade-off studies, and prototyping.

3.3850

software requirements management

1. process of planning and controlling the identification, allocation, and flow-down of requirements from the system level to the module or part level, including interfaces, verification, modifications, and status monitoring
cf. requirements management

3.3851

software requirements phase

1. software development life-cycle phase during which the requirements for a software product, such as functional and performance capabilities, are defined, documented, and reviewed

3.3852

software requirements review (SRR)

1. review of the requirements specified for one or more software configuration items to evaluate their responsiveness to and interpretation of the system requirements and to determine whether they form a satisfactory basis for proceeding into preliminary design of the configuration items **2.** review as in (1) for any software component

cf. system requirements review

Note 1 to entry: This review is called software specification review by the US Department of Defense.

3.3853

software requirements specification (SRS)

1. documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.29*] **2.** structured collection of the requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.24*]

3.3854

software requirements verification

1. process of ensuring that the software requirements specification complies with the system requirements, conforms to document standards of the requirements phase, and is an adequate basis for the architectural (preliminary) design phase
cf. requirements verification, requirements validation

3.3855

software safety

1. freedom from software hazards [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans*, 3.1.6]
cf.: system safety

3.3856

software safety program

1. systematic approach to reducing software risks [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans*, 3.1.7]

3.3857

software system

1. system for which software is of primary importance to the stakeholders
cf. software-intensive system

3.3858

software test environment (STE)

1. facilities, hardware, software, firmware, procedures, and documentation needed to perform qualification or other testing of software [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.15]

Note 1 to entry: Elements include simulators, code analyzers, test case generators, path analyzers, and elements used in the software engineering environment

3.3859

software test incident

1. event occurring during the execution of a software test that requires investigation.

3.3860

software testing

1. activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2] 2. dynamic verification that a program provides expected behaviors on a finite set of test cases, suitably selected from the usually infinite executions domain [*ISO/IEC TR 19759:2016 Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOk)*, 4]

3.3861

software testing environment

1. facilities, hardware, software, firmware, procedures, and documentation needed to perform testing of software [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*]

3.3862

software tool

1. computer program used in the development, testing, analysis, or maintenance of a program or its documentation 2. software product providing automatic support for software life-cycle tasks [*ISO/IEC TR 15846:1998 Information technology — Software life cycle processes — Configuration Management*, 4.6]

EXAMPLE: comparator, cross-reference generator, decompiler, driver, editor, flowchart, monitor, test case generator, timing analyzer

3.3863

software transition

1. controlled and coordinated sequence of actions wherein software development passes from the organization performing initial software development to the organization performing software maintenance [*IEEE 14764-2006 Software Engineering - Software Life Cycle Processes - Maintenance, 3.10*]

3.3864

software unit

1. atomic-level software component of the software architecture that can be subjected to standalone testing [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.49*] **2.** separately compilable piece of code [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.43*]

3.3865

software usage

1. consumption against a software entitlement measured as defined by the terms and conditions of that entitlement [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.47*]

Note 1 to entry: Depending on the specific terms and conditions, usage can include accessing, copying, distributing, installing and executing software.

3.3866

software user documentation

1. electronic or printed body of material that provides information to users of software

3.3867

software version ID

software version identification

1. explicit and immutable version identifier (name or number) inserted into each configuration item, including each individual release, that can be used to identify the exact version of the configuration item in any instance or repository [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*]

3.3868

software-based system

1. computer system controlled by software.

3.3869

software-intensive system

1. system for which software is a major technical challenge and is perhaps the major factor that affects system schedule, cost, and risk [*IEEE 1062-2015 IEEE Recommended Practice for Software Acquisition, 3.1*] cf. software system

Note 1 to entry: In the most general case, a software-intensive system is comprised of hardware, software, people, and manual procedures.

3.3870

software/system element

1. element that defines and prescribes what a software or system is composed of [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities, 3.12*] cf. software element

EXAMPLE: requirements, design, code, test cases, and version number

Note 1 to entry: An element can contain sub elements or other software/system elements that are dependent on the top-level element.

3.3871

SOI

1. system-of-interest [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.2*]

428

3.3872

SOIC

1. small outline integrated circuit

3.3873

solution domain

1. environment in which a solution or set of solutions resides
cf. problem domain

3.3874

SOO

1. statement of objectives [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.2*]

3.3875

SOP

1. standard operating procedure **2.** small outline package

3.3876

SoPC

1. system on a programmable chip

3.3877

sort

1. symbol representing the name of a set [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.23*]

3.3878

sorting

1. activity of sequencing of rows or records in a transactional function [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.48*]

3.3879

SoS

1. system of systems [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.2*]

3.3880

source

1. item or activity having a potential for a consequence [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management, 3.18*]

Note 1 to entry: In the context of safety, source is a hazard (refer to ISO/IEC Guide 51:2014).

3.3881

source address

1. address of a device or storage location from which data is to be transferred
cf. destination

3.3882

source code

1. computer instructions and data definitions expressed in a form suitable for input to an assembler, compiler, or other translator
cf. object code

Note 1 to entry: A source program is made up of source code

3.3883

source code organization

1. arrangement of source code, including layout of code within a single file and packaging of source code into modules, classes, physical files, and so on

3.3884

source language

1. language in which the input to a machine-aided translation process is represented

cf. target language

EXAMPLE: the language used to write a computer program

3.3885

source node

1. node associated with the start of an arc [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.17]

3.3886

source program

1. computer program that must be compiled, assembled, or otherwise translated in order to be executed by a computer

cf. object program

3.3887

source selection criteria

1. a set of attributes desired by the buyer which a seller must meet or exceed to be selected for a contract [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3888

source statements (SS)

1. encoded logic of the software product

Note 1 to entry: Source statements can be classified by function as executable, data declaration, compiler directive, or comment. They can also be classified as deliverable or nondeliverable.

3.3889

SOW

1. statement of work [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition; 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]

3.3890

spawn action

1. dividing action, where the enabled chains will not join [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.1.6]

3.3891

special cause

assignable cause

1. source of variation that is not inherent in the system, is not predictable, and is intermittent

cf. common cause

EXAMPLE: It can be assigned to a defect in the system. On a control chart, it is indicated by points beyond the control limits or non-random patterns within the control limits.

3.3892

special function register (SFR)

1. register used to control or monitor functions in a microprocessor

3.3893

specialize

1. change by an instance from being an instance of its current class to being additionally an instance of one (or more) of the subclasses of the current subclass [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.181]

cf. respecialize; unspecialize

Note 1 to entry: A specialized instance acquires a different (lower) lowclass.

3.3894

specific symbol

1. symbol used when the precise nature or form of, for example, the process or data media is known and when it is necessary to depict the actual medium [*ISO 5807:1985 Information processing — Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts, 3.2*]

3.3895

specification

1. detailed formulation, in document form, which provides a definitive description of a system for the purpose of developing or validating the system [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** information item that identifies, in a complete, precise, and verifiable manner, the requirements, design, behavior, or other expected characteristics of a system, service, or process [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.24*] **3.** a document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system, component, product, result, or service and, often, the procedures for determining whether these provisions have been satisfied. Examples are: requirement specification, design specification, product specification, and test specification [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **4.** concrete representation of a model in some notation [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 7.4*]

3.3896

specification change notice (SCN)

1. a document used in configuration management to propose, transmit, and record changes to a specification
cf. configuration control, engineering change, notice of revision

3.3897

specification language

1. a language, often a machine-processible combination of natural and formal language, used to express the requirements, design, behavior, or other characteristics of a system or component
cf. programming language, query language

EXAMPLE: a design language or requirements specification language

3.3898

specification limits

1. the area, on either side of the centerline, or mean, of data plotted on a control chart that meets the customer's requirements for a product or service. This area may be greater than or less than the area defined by the control limits [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. control limits

3.3899

specification tree

1. diagram that depicts the specifications for a given system and shows their relationships to one another.

3.3900

specification-based testing

black-box testing

closed-box testing

1. testing in which the principal test basis is the external inputs and outputs of the test item, commonly based on a specification, rather than its implementation in source code or executable software [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.39*]
cf. functional testing

3.3901

SPI

1. schedule performance index [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** serial peripheral interface **3.** system/software process improvement [*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, 4.2*]

3.3902

spiral model

1. model of the software development process in which the constituent activities, typically requirements analysis, preliminary and detailed design, coding, integration, and testing, are performed iteratively until the software is complete

cf. waterfall model, incremental development, rapid prototyping

3.3903

SPLC

1. software project life cycle.

3.3904

SPLCM

1. software project life cycle model.

3.3905

SPLCP

1. software project life cycle process.

3.3906

split key

1. foreign key containing two or more attributes, where at least one of the attributes is a part of the entity's primary key and at least one of the attributes is not a part of the primary key [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.183]

Note 1 to entry: [key style]

3.3907

SPMPI

1. Software Project Management Planned Information.

3.3908

sponsor

1. a person or group who provides resources and support for the project, program, or portfolio and is accountable for enabling success [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3909

sponsoring organization

1. the entity responsible for providing the project's sponsor and a conduit for project funding or other project resources [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3910

spool

1. to read input data, or write output data, to auxiliary or main storage for later processing or output, in order to permit input/output devices to operate concurrently with job execution

Note 1 to entry: derived from the acronym SPOOL for Simultaneous Peripheral Output On Line

3.3911

spooler

1. program that initiates and controls spooling

3.3912

SPQM-RM

1. software product quality measurement reference model [*ISO/IEC 25020:2007 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Measurement reference model and guide*, 5]

3.3913

spreadsheet program

1. program that displays a table of cells arranged in rows and columns, in which the change of the contents of one cell can cause recomputation of one or more cells based on user-defined relations among the cells [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

ISO/IEC/IEEE 24765:2017(E)

3.3914

sprint

- 1.** short time frame, in which a set of software features is developed, leading to a working product that can be demonstrated to stakeholders [*ISO/IEC/IEEE 26515: 2011 Systems and software engineering: Developing user documentation in an agile environment*, 4.14]

Note 1 to entry: In some organizations, a sprint is known as an iteration.

3.3915

SPS

- 1.** system performance specification [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.1]

3.3916

SQA

- 1.** Software Quality Assurance [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.3]

3.3917

SQAP

- 1.** software quality assurance plan [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.3]

3.3918

SQC

- 1.** software quality control [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.2]

3.3919

SQL

- 1.** Structured Query Language

3.3920

squiggle

- 1.** short "S"-shaped line attached at one end to an arrow label and at the other end to an arrow segment [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.101]

Note 1 to entry: A squiggle binds an object type set (arrow label) to an object set (arrow segment).

3.3921

sr

- 1.** system definition and realization [*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]

3.3922

SRAM

- 1.** static random access memory

3.3923

SRD

- 1.** system requirements document [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]

3.3924

SRMPI

- 1.** Software Release Management Planned Information.

3.3925

SRR

1. software requirements review **2.** system requirements review [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]
cf. SAR

3.3926

SRS

1. software requirements specification **[IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2]** **2.** system requirements specification.
cf. SyRS

3.3927

SS

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

3.3928

SSE

1. system security engineering [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.3929

SSL

1. secure sockets layer [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3*]

3.3930

SSOP

1. shrink small outline package

cf. SOIC

3.3931

SSR

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

cf. software requirements review

3.3932

SSS

1. system/subsystem specification [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.1*]

3.3933

stability

1. property that an object has with respect to a given failure mode if it cannot exhibit that failure mode [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.6.4*]

3.3934

stability schema

1. specification of failure modes which an object will not exhibit [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 16.2.1.1*]

3.3935

stabilization phase

1. time interval of the measurement procedure when the RTE starts submitting tasks until the SUT reaches a stable state of operation [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.16*]

3.3936

stable branch

1. branch where stability-disrupting changes are discouraged

Note 1 to entry: the branch used for releasing the product's stable production version.

3.3937

stable process

- 1.** process from which all special causes of process variation have been removed and prevented from recurring, so that only common causes of process variation of the process remain

3.3938

stack pointer

- 1.** register that stores the address at the top of a stack (the address of the most recent program request)

3.3939

staff-hour

- 1.** hour of effort expended by a member of the staff

3.3940

staffing management plan

- 1.** a component of the human resource plan that describes when and how project team members will be acquired and how long they will be needed [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3941

stage

- 1.** period within the life cycle of an entity that relates to the state of its description or realization [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.44; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.50; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.43]
cf. project phase

Note 1 to entry: Stages relate to major progress and achievement milestones of the system through its life cycle. Stages can overlap.

3.3942

staged representation

- 1.** structure wherein attaining the goals of a set of process areas establishes a maturity level; each level builds a foundation for subsequent levels

3.3943

stakeholder

- 1.** individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.51; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.44] **2.** an individual, group, or organization who may affect, be affected by, or perceive itself to be affected by a decision, activity, or outcome of a project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **3.** [system] individual, team, organization, or classes thereof, having an interest in a system [*ISO/IEC/IEEE 42010:2011 Systems and software engineering — Architecture description*, 3.10] **4.** individual, group or organization that can affect, be affected by, or perceive itself to be affected by, a risk [*ISO/IEC 16085:2006 Systems and software engineering — Life cycle processes — Risk management*, 3.19]

EXAMPLE: end users, end user organizations, supporters, developers, producers, trainers, maintainers, disposers, acquirers, supplier organizations and regulatory bodies, interested parties, decision-makers

Note 1 to entry: Some stakeholders can have interests that oppose each other or oppose the system.

3.3944

stakeholder analysis

- 1.** a technique of systematically gathering and analyzing quantitative and qualitative information to determine whose interests should be taken into account throughout the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3945

stakeholder equity

- 1.** degree of the share or claim a stakeholder has in the system of interest or a portion of the system of interest [*ISO/IEC TR 24766:2009 Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities*, 3.9]

3.3946

stakeholder management plan

- 1.** The stakeholder management plan is a subsidiary plan of the project management plan that defines the processes, procedures, tools and techniques to effectively engage stakeholders in project decisions and execution based on the analysis of their needs, interests and potential impact. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3947

stakeholder register

- 1.** a project document including the identification, assessment and classification of project stakeholders [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3948

stand-alone

- 1.** pertaining to hardware or software that is capable of performing its function without being connected to other components

EXAMPLE: a stand-alone document processing system

3.3949

standard

- 1.** document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.33] **2.** a document that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. guide, guideline

EXAMPLE: [ISO/IEC TR 10000-1]

3.3950

standard process

- 1.** set of definitions of the processes used to guide processes in an organization [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.1.8]

Note 1 to entry: These process definitions cover the fundamental process elements (and their relationships to each other) that must be incorporated into the defined processes that are implemented in projects across the organization. A standard process establishes consistent activities across the organization and is desirable for long-term stability and improvement. The organization's set of standard processes describes the fundamental process elements that will be part of the projects' defined processes. It also describes the relationships (for example, ordering and interfaces) between these process elements.

3.3951

standardized profile

- 1.** internationally agreed-to, harmonized standard which describes one or more profiles [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.12]

3.3952

standby mode

sleep mode

- 1.** operating mode which saves power when a microcontroller unit is not in active use

3.3953

standby redundancy

1. in fault tolerance, the use of redundant elements that are left inoperative until a failure occurs in a primary element

cf. active redundancy

3.3954

start date

1. a point in time associated with a schedule activity's start, usually qualified by one of the following: actual, planned, estimated, scheduled, early, late, target, baseline, or current [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3955

start-to-finish (SF)

1. a logical relationship in which a successor activity cannot finish until a predecessor activity has started [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. logical relationship

3.3956

start-to-start (SS)

1. a logical relationship in which a successor activity cannot start until a predecessor activity has started [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

cf. logical relationship

3.3957

starting address

1. address of the first instruction of a computer program in main storage

cf. origin, assembled origin, loaded origin

Note 1 to entry: This address can be the same as the program's origin, depending upon whether there are data preceding the first instruction.

3.3958

state

1. condition or mode of existence that a system, component, or simulation can be in **2.** values assumed at a given instant by the variables that define the characteristics of a system, component, or simulation **3.** unique value that represents the stage of progress of software in its execution [*ISO/IEC 11411:1995 Information technology — Representation for human communication of state transition of software, 2.1*] **4.** condition that characterizes the behavior of a function, subfunction or element at a point in time **5.** at a given instant in time, the condition of an object that determines the set of all sequences of actions (or traces) in which the object can participate [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 8.8*] **6.** identified condition or set of conditions within the subject software that is associated with certain of the possible actions of the subject software [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*] **7.** characteristic of a unit (usually composite in structure and multi-dimensional) expressing the cumulative effect of all previous unit interaction occurrences, in terms of which the delayed affective relationships of the unit can be evaluated [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.25*]

Note 1 to entry: A state is a shorthand representation for the unit's interaction occurrence history. Ascribing "state" to a unit implies that it has a capability to modify future behaviors as a result of past interactions with its environment.

3.3959

state class

1. class that represents a set of real or abstract objects that have common knowledge or behavior [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.184*]

EXAMPLE: people, places, events, ideas, things, combinations of things

Note 1 to entry: A state class represents instances with changeable state. The constituent instances of a state class can come and go and can change state over time; that is, their property values can change.

3.3960

state data

- 1.** data that defines an internal state of the test unit and is used to establish that state or compare with existing states.

3.3961

state diagram

- 1.** diagram that depicts the states that a system or component can assume, and shows the events or circumstances that cause or result from a change from one state to another

3.3962

state invariant condition

- 1.** statement of constraints or relations that can be used to distinguish a subset of particular property states that all satisfy the condition [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.26]

3.3963

state name

- 1.** unique identifier of the state of software execution [*ISO/IEC 11411:1995 Information technology — Representation for human communication of state transition of software*, 2.1]

3.3964

statement

- 1.** in a programming language, a meaningful expression that defines data, specifies program actions, or directs the assembler or compiler

cf. assignment statement, control statement, declaration

3.3965

statement coverage

- 1.** percentage of the set of all executable statements of a test item that are covered by a test set [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.40]

3.3966

statement of work (SOW)

- 1.** document used by the acquirer to describe and specify the tasks to be performed under the contract [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.46] **2.** a narrative description of products, services, or results to be delivered by the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **3.** document used by the acquirer that includes the needs and expectations, the scope, objectives and deliverables [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile*, 3.11] **4.** means to describe and specify the tasks to be performed under the contract [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.65]

3.3967

statement testing

- 1.** testing designed to execute each statement of a computer program **2.** test design technique in which test cases are constructed to force execution of individual statements in a test item [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.41]

cf. branch testing, path testing

3.3968

StateTransition

- 1.** change from one State to another [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.3969

static

- 1.** pertaining to an event or process that occurs without computer program execution

cf. dynamic

EXAMPLE: static analysis, static binding

3.3970

static analysis

- 1.** process of evaluating a system or component based on its form, structure, content, or documentation
cf. dynamic analysis, inspection, walk-through

3.3971

static binding

- 1.** binding performed prior to the execution of a computer program and not subject to change during program execution
cf. dynamic binding

3.3972

static breakpoint

- 1.** breakpoint that can be set at compile time, such as entry into a given routine
cf. dynamic breakpoint, code breakpoint, data breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint

3.3973

static dump

- 1.** dump that is produced before or after the execution of a computer program
cf. dynamic dump, change dump, memory dump, postmortem dump, selective dump, snapshot dump

3.3974

static error

- 1.** error that is independent of the time-varying nature of an input
cf. dynamic error

3.3975

static model

- 1.** model that describes an interrelated set of classes (and/or subject domains) along with their relationships and responsibilities [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.185*]
cf. dynamic model

3.3976

static product

- 1.** non-executable system or software product for reviewing [*ISO/IEC 25041: 2012 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation guide for developers, acquirers and independent evaluators, 4.12*]

3.3977

static random access memory (SRAM)

- 1.** random access memory without refresh process, which keeps data as long as it is powered on

Note 1 to entry: based on a circuit architecture with double stable states

3.3978

static schema

- 1.** specification of the state of one or more information objects, at some point in time, subject to the constraints of any invariant schemata [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 6.1.2*]

Note 1 to entry: Thus, a static schema is the specification of the types of one or more information objects at some particular point in time. These types are subtypes of the types specified in the invariant schema.

3.3979

static testing

1. testing in which a test item is examined against a set of quality or other criteria without code being executed
[ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.42]
cf. inspection

EXAMPLE: reviews, static analysis.

3.3980

statistical process control

1. statistically based analysis of a process and measures of process performance, which identify common and special causes of variation in process performance and maintain process performance within limits

3.3981

statistical sampling

1. choosing part of a population of interest for inspection [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.3982

statistically managed process

1. process that is managed by a statistically based technique in which processes are analyzed, special causes of process variation are identified, and performance is contained within well-defined limits

3.3983

status code

condition code

1. code used to indicate the results of a computer program operation

EXAMPLE: a code indicating a carry, an overflow, or a parity error

3.3984

STE

1. software test environment [IEEE 730-2014 *IEEE Standard for Software Quality Assurance Processes*, 3.3]

3.3985

step

1. one element (numbered list item) in a procedure that tells a user to perform an action (or actions) [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.47] 2. simultaneous occurrence of a finite multiset of transition modes that are concurrently enabled in a marking [ISO/IEC 15909-1:2004 *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26.4] 3. abstraction of an action, used in a process, which can leave unspecified some or all of the objects that participate in that action [ISO/IEC 15414:2015 *Information technology — Open distributed processing — Reference model — Enterprise language*, 6.3.6]

Note 1 to entry: A step contains one or more actions. Responses by the software are not considered to be steps.

3.3986

stepwise refinement

1. software development technique in which data and processing steps are defined broadly at first and then further defined with increasing detail
cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, structured design, transaction analysis, transform analysis

3.3987

stimulus

1. whatever causes a unit to exhibit an occurrence of a behavior pattern in the unit's repertoire; something causing or regarded as causing a response [IEEE 1175.4-2008 *IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*, 3.27]

Note 1 to entry: A stimulus can be the occurrence of an externally controlled interaction affecting the unit (external stimulus), or it can be the occurrence of an internally controlled event (internal stimulus).

3.3988

STK

1. stakeholder [*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*]

3.3989

STL

1. semantic transfer language [*IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*]

3.3990

stock keeping unit (sku)

1. identification, usually alphanumeric, of a particular product that allows it to be tracked for inventory and software entitlement purposes [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.48*]

Note 1 to entry: typically associated with unique products for sales purposes, such as software entitlements. It can correspond uniquely to specific software products, or represent packages of software, with specific terms and conditions, such as whether it relates to a full product, upgrade product, or maintenance on an existing product.

3.3991

stop

1. to terminate the execution of a computer program

cf. pause

3.3992

storage

storage device

1. functional unit into which data can be placed, in which data can be retained, and from which data can be retrieved [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.3993

storage allocation

1. element of computer resource allocation, consisting of assigning storage areas to specific jobs and performing related procedures, such as transfer of data between main and auxiliary storage, to support the assignments made
cf. buffer, contiguous allocation, cyclic search, memory compaction, overlay, paging, virtual storage

3.3994

storage capacity

1. maximum number of items that can be held in a given storage device; usually measured in words or bytes
cf. channel capacity, memory capacity

3.3995

storage efficiency

1. degree to which a system or component performs its designated functions with minimum consumption of available storage

cf. execution efficiency

3.3996

store

1. to place or retain data in a storage device **2.** to copy computer instructions or data from a register to internal storage or from internal storage to external storage

3.3997

story point

1. the relative measure of the effort needed to develop a user story, compared with what is considered a typical user story by the project team [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.3998

STP

1. software test plan [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.3999

straight-line code

1. sequence of computer instructions in which there are no loops

3.4000

straight-line coding

1. programming technique in which loops are avoided by stating explicitly and in full all of the instructions that would be involved in the execution of each loop

cf. unwind

3.4001

strategy

1. organization's overall plan of development, describing the effective use of resources in support of the organization in its future activities

Note 1 to entry: involves setting objectives and proposing initiatives for action

3.4002

stratified language

1. language that cannot be used as its own metalanguage

cf. unstratified language

EXAMPLE: FORTRAN, COBOL

3.4003

stream interface

1. interface in which all the interactions are flows [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.8*]

3.4004

stream interface signature

1. interface signature for a stream interface [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 7.1.13*]

Note 1 to entry: A stream interface comprises a finite set of action templates, one for each flow type in the stream interface. Each action template for a flow contains the name of the flow, the information type of the flow, and an indication of causality for the flow (i.e. producer or consumer but not both) with respect to the object which instantiates the template. The phrase "complementary interface signature to X", where X is itself an interface signature describes an interface signature identical to X in all respects except causality, which is opposite to that in X. Many Interface Definition Languages (IDLs) capture only the action templates of a signature and depend upon the context in which the IDL is used to determine the causality that is to be applied.

3.4005

strengths, weaknesses, opportunities, and threats (SWOT) analysis

1. analysis of strengths, weaknesses, opportunities and threats of an organization, project, or option [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4006

stress testing

1. type of performance efficiency testing conducted to evaluate a test item's behavior under conditions of loading above anticipated or specified capacity requirements, or of resource availability below minimum specified requirements [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.43*]

cf. boundary value

3.4007

strong typing

1. feature of some programming languages that requires the type of each data item to be declared, precludes the application of operators to inappropriate data types, and prevents the interaction of data items of incompatible types

3.4008

STRS

1. stakeholder requirements specification [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

3.4009

structural testing

glass-box testing

white-box testing

1. testing that takes into account the internal mechanism of a system or component
cf. functional testing (1), structure-based testing

Note 1 to entry: Types include branch testing, path testing, statement testing.

3.4010

structure chart

hierarchy chart

program structure chart

1. diagram that identifies modules, activities, or other entities in a system or computer program and shows how larger or more general entities break down into smaller, more specific entities
cf. call graph

Note 1 to entry: The result is not necessarily the same as that shown in a call graph.

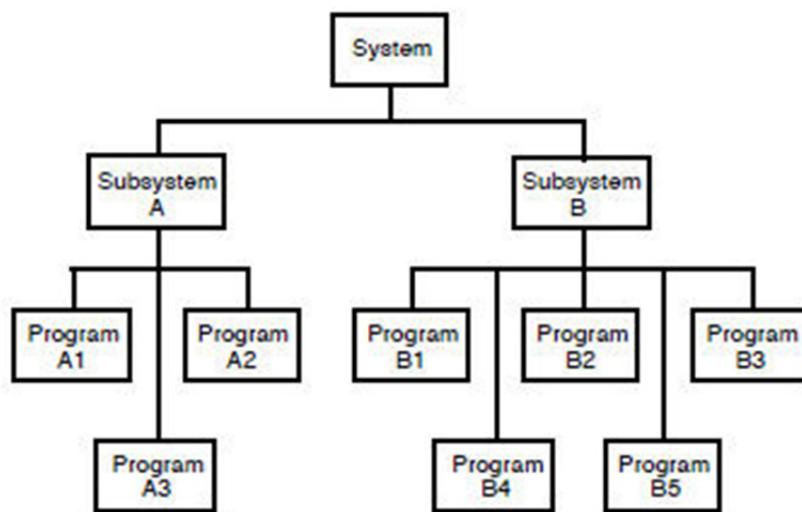


Figure 17 — Structure chart

3.4011

structure clash

1. in software design, a situation in which a module must deal with two or more data sets that have incompatible data structures

cf. data structure-centered design, order clash

3.4012

structure-based testing

1. dynamic testing in which the tests are derived from an examination of the structure of the test item [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.46]
cf. structural testing

Note 1 to entry: Structure-based testing is not restricted to use at component level and can be used at all levels, e.g. menu item coverage as part of a system test.

3.4013

structured design

1. disciplined approach to software design that adheres to specified rules based on principles such as modularity, top-down design, and stepwise refinement of data, system structures, and processing steps **2.** result of applying the approach in (1)

cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping.

3.4014

structured program

1. computer program constructed of a basic set of control structures, each having one entry and one exit
cf. structured design

Note 1 to entry: The set of control structures typically includes: sequence of two or more instructions, conditional selection of one of two or more sequences of instructions, and repetition of a sequence of instructions.

3.4015

structured programming

1. software development technique that includes structured design and results in the development of structured programs

3.4016

structured programming language

1. programming language that provides the structured program constructs, namely, single-entry-single-exit sequences, branches, and loops, and facilitates the development of structured programs

cf. block-structured language

3.4017

structured walkthrough

1. systematic examination of the requirements, design, or implementation of a system, or any part of it, by qualified personnel [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4018

stub

1. skeletal or special-purpose implementation of a software module, used to develop or test a module that calls or is otherwise dependent on it **2.** computer program statement substituting for the body of a software module that is or will be defined elsewhere **3.** engineering object in a channel, which interprets the interactions conveyed by the channel, and performs any necessary transformation or monitoring based on this interpretation [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 8.1.9] **4.** scaffolding code written for the purpose of exercising higher-level code before the lower-level routines that will ultimately be used are available

3.4019

style

1. set of language-specific editorial conventions covering grammar, terminology, punctuation, capitalization, and word choice of documentation [*ISO/IEC 26514:2008 Systems and software engineering — Requirements for designers and developers of user documentation*, 4.48]

3.4020

sub-path

1. path that is part of a larger path [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.26]

3.4021

subactivity

1. subgraph of an activity which is itself an activity and which satisfies the following condition: for any pair of fork-join actions in the parent activity, if one of these actions is included in the subgraph, then both must be included in the subgraph [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.1.8]

3.4022

subclass

1. specialization of one or more superclasses [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.186] **2.** relation between class A and class B in which the type associated with A is a subtype of the type associated with B [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.12]
cf. subtype, superclass

Note 1 to entry: Each instance of a subclass is an instance of each superclass. A subclass typically specifies additional, different responsibilities to those of its superclasses or overrides superclass responsibilities to provide a different realization.

3.4023

subclass cluster

category cluster

1. set of one or more generalization structures in which the subclasses share the same superclass and in which an instance of the superclass is an instance of no more than one subclass [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.187] **2.** set of one or more mutually exclusive specializations of the same generic entity [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.187]

Note 1 to entry: A cluster exists when an instance of the superclass can be an instance of only one of the subclasses in the set, and each instance of a subclass is an instance of the superclass.

3.4024

subclass responsibility

1. designation that a property of a class must be overridden in its subclasses [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.188]

Note 1 to entry: That is, the designation given to a property whose implementation is not specified in this class. A property that is a subclass responsibility is a specification in the superclass of an interface that each of its subclasses must provide. A property that is designated as a subclass responsibility has its realization deferred to the subclass(es) of the class.

3.4025

subdomain

1. domain which is a subset of a given domain [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 10.4]

3.4026

subject area

1. related collection of meta-object instance definitions [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: Subject areas are used to define scoped areas of interest. Subject areas overlap to ensure the integration of the overall metamodel, but a tool need only use those subject areas relevant to the data to be exported or imported.

3.4027

subject domain

1. area of interest or expertise [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.189]

Note 1 to entry: The responsibilities of a subject domain are an aggregation of the responsibilities of a set of current or potential named classes. A subject domain can also contain other subject domains. A subject domain encapsulates the detail of a view.

3.4028

subject domain responsibility

1. generalized concept that the analyst discovers by asking, "In general, what do instances in this subject domain need to be able to do or to know?" [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.190]

Note 1 to entry: The classes and subject domains in a subject domain together supply the knowledge, behavior, and rules that make up the subject. These notions are collectively referred to as the subject domain's responsibilities. Subject domain responsibilities are not distinguished as sub-domains or classes during the early stages of analysis.

3.4029

subject software

1. computing (software) application (existing or to be created) about which descriptive information is being developed in a computing system tool or a Computer-Aided Software Engineering (CASE) tool [IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior]

3.4030

subject system

1. computing system (existing or to be created) about which descriptive information is being developed in a computing system or CASE tool [IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.1; IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.13]

3.4031

subject tool

1. particular computing system tool or CASE tool that is the focus for a description of interconnections and tool content [IEEE 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description, 3.11; IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.14]

3.4032

subject unit

1. unit that is the subject of a behavior modeling [IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior, 3.28]

Note 1 to entry: A subject unit is segregated from a collection of units that constitute the environment in which its behaviors are solicited and exhibited.

3.4033

submit primitive

1. service primitive for which the protocol object is the initiating object of the corresponding communication [ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 3.3.12]

3.4034

subnetwork

1. a subdivision (fragment) of a project schedule network diagram, usually representing a subproject or a work package. Often used to illustrate or study some potential or proposed schedule condition, such as changes in preferential schedule logic or project scope [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4035

subprogram

1. separately compilable, executable component of a computer program
cf. coroutine, main program, routine, subroutine

Note 1 to entry: The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages.

3.4036

subproject

1. a smaller portion of the overall project created when a project is subdivided into more manageable components or pieces [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4037

subroutine

1. routine that returns control to the program or subprogram that called it

cf. coroutine, closed subroutine, open subroutine

Note 1 to entry: The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages.

3.4038

subroutine trace

call trace

1. record of all or selected subroutines or function calls performed during the execution of a computer program and, optionally, the values of parameters passed to and returned by each subroutine or function

cf. execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

3.4039

subscription-based license

service-based license

term-based license

1. license for an entitlement that is for a limited amount of time [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.1.39*]

Note 1 to entry: It is not a perpetual license and requires renewal to remain in force.

3.4040

substitutability

1. principle stating that, since each instance of a subclass is an instance of the superclass, an instance of the subclass is acceptable in any context where an instance of the superclass is acceptable [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.191*]

Note 1 to entry: Any request sent to an instance receives an acceptable response, regardless of whether the receiver is an instance of the subclass or the superclass.

3.4041

subsystem

1. secondary or subordinate system within a larger system

3.4042

subtype

1. subset of a data type, obtained by constraining the set of possible values of the data type **2.** meta-entity that inherits all of the meta-attributes and meta-relationships of its immediate and indirect supertype meta-entities [*ISO/IEC 15474-2:2002 Information technology — CDIF framework — Part 2: Modelling and extensibility, 6.2.5*] **3.** relation of type A to type B when every <X> which satisfies A also satisfies B [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.11*]
cf. derived type, supertype

Note 1 to entry: The operations applicable to the subtype are the same as those of the original data type.

3.4043

success criteria

1. set of conditions to be satisfied by a process instance at completion [*ISO/IEC 30103:2015 Software and Systems Engineering - Lifecycle Processes - Framework for Product Quality Achievement, 3.9*]

Note 1 to entry: Information items and artefacts produced by the process instance must meet the success criteria. Success criteria are established based on the outcomes of the corresponding life cycle process, requirements of the system element to which the process instance contributes, and requirements and constraints arising from decisions in other process instances.

3.4044

successful adoption

1. extent to which the use of CASE tools can measurably meet an organization's uniquely defined adoption goals [*ISO/IEC TR 14471:2007 Information technology — Software engineering — Guidelines for the adoption of CASE tools*, 2.1.1]

3.4045

sunk cost

1. cost that is irrecoverable by future actions

Note 1 to entry: Sunk costs have a psychological impact, but are irrelevant in business decisions.

3.4046

superclass

1. class whose instances are specialized into one or more subclasses [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.193] **2.** relation between class B and class A, when the type associated with A is a subtype of the type associated with B [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.12]
cf. generic entity, partial cluster, total cluster, subclass, supertype

3.4047

supercomputer

1. class of computers that have the highest processing speeds available at a given time [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: often used for solving scientific and engineering problems

3.4048

supertype

1. relation between type B and type A, in which every <X> which satisfies A also satisfies B [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.11]
cf. generic entity, subtype, superclass

3.4049

supervisor state

executive state

master state

privileged state

1. in the operation of a computer system, a state in which the supervisory program is executing
cf. problem state

Note 1 to entry: This state usually has higher priority than, and precludes the execution of, application programs.

3.4050

supervisory program

control program

executive

executive program

supervisor

1. computer program, usually part of an operating system, that controls the execution of other computer programs and regulates the flow of work in a computer system
cf. supervisor state

3.4051

supplemental Ent

supplemental entitlement schema

1. Ent which has an <entType> of Supplemental [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.40]

Note 1 to entry: Supplemental Ents provide extended information about a primary Ent and are linked to primary Ents by the <linkedToPrimaryEntId> attribute.

3.4052

supplementary run

1. time interval of the measurement procedure from the time the measurement results fulfill the required statistical significance to the time when all tasks, which were submitted during the rating interval, are completed [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.17*]

3.4053

supplier

contractor

producer

vendor

1. organization or individual that enters into an agreement with the acquirer for the supply of a product or service [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.47; ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.37; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.52; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.45*] **2.** individual or organization that enters into a contract with the acquirer for the supply of a system, software product or software service under the terms of the contract [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.37; ISO/IEC 25040:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process, 4.63*] **3.** organization or part of an organization or individual that enters into an agreement with the application management organization for the supply of a product, service, materials, or human capacity [*ISO/IEC 16350-2015 Information technology — Systems and software engineering — Application management, 4.34*]

Note 1 to entry: The acquirer and the supplier sometimes are part of the same organization. The application management organization can have internal or external suppliers. A supplier can be another application management organization, but also IT infrastructure management organizations or consultants.

3.4054

support

1. set of activities necessary to ensure that an operational system or component fulfills its original requirements and any subsequent modifications to those requirements

cf. software life cycle, system life cycle

EXAMPLE: software or hardware maintenance, user training

3.4055

support activity group

1. activity group that is necessary to assure the successful completion of a project, but consists of supporting activities rather than activities directly oriented to the development effort

3.4056

support manual

1. document that provides the information necessary to service and maintain an operational system or component throughout its life cycle

cf. diagnostic manual, installation manual, maintenance manual, operator manual, programmer manual, user manual

Note 1 to entry: Typically described are the hardware and software that make up the system or component and procedures for servicing, repairing, or reprogramming it.

3.4057

support software

1. software that aids in the development or maintenance of other software **2.** software or a program that aids in the development, maintenance, or use of other software or provides general application-independent capability [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. application software, system software

EXAMPLE: compilers, loaders, and other utilities

3.4058

support staff-hour

- 1.** hour of effort expended by a member of the staff who does not directly define or create the software product, but acts to assist those who do

3.4059

surveillance

- 1.** systematic iteration of conformity assessment activities as a basis for maintaining the validity of the statement of conformity [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.57]

3.4060

survivability

- 1.** degree to which a product or system continues to fulfill its mission by providing essential services in a timely manner in spite of the presence of attacks [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.6]
cf. recoverability

3.4061

suspension criteria

- 1.** criteria used to (temporarily) stop all or a portion of the testing activities [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.46]

3.4062

sustainment

- 1.** activities performed to ensure that a product or service remains operational
cf. maintenance

3.4063

SV

- 1.** schedule variance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4064

SVD

- 1.** software version description [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.2]

3.4065

SVR

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

3.4066

SW

- 1.** software [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]

3.4067

swap

- 1.** exchange of the contents of two storage areas, usually an area of main storage with an area of auxiliary storage
2. to perform an exchange as in (1)
cf. roll in, roll out

3.4068

SWEBOK

- 1.** Software Engineering Body of Knowledge [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management*, 3]

3.4069

SWID

- 1.** software identification [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.40]

3.4070

SWOT

1. strengths, weaknesses, opportunities, and threats [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4071

symbol

1. graphic representation of a concept that has meaning in a specific context [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4072

symbol table

1. table that presents program symbols and their corresponding addresses, values, and other attributes

3.4073

symbolic address

1. address expressed as a name or label that must be translated to the absolute address of the device or storage location to be accessed

cf. absolute address

3.4074

symbolic execution

1. software analysis technique in which program execution is simulated using symbols, such as variable names, rather than actual values for input data, and program outputs are expressed as logical or mathematical expressions involving these symbols

3.4075

symbolic language

1. programming language that expresses operations and addresses in symbols convenient to humans rather than in machine language

cf. machine language

EXAMPLE: assembly language, high order language

3.4076

symbolic trace

1. record of the source statements and branch outcomes that are encountered when a computer program is executed using symbolic, rather than actual, values for input data

cf. execution trace, retrospective trace, subroutine trace, variable trace

3.4077

synchronize

1. to pull the changes made in a parent branch into its (evolving) child (for example, feature) branch **2.** to update a view with the current version of the files in its corresponding branch

3.4078

synchronous

1. pertaining to two or more processes that depend upon the occurrence of specific events such as common timing [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4079

synchronous dynamic random access memory (SDRAM)

1. DRAM with memory access driven by a clock that is synchronized with the processor's memory bus clock

Note 1 to entry: SDRAM can access data fractions in different memory banks simultaneously.

3.4080

synchronous graphics random access memory (SGRAM)

1. SDRAM designed for the graphics card of a computer

3.4081

synchronous message communication

tightly coupled message communication

- 1.** form of communication in which a producer task sends a message to a consumer task and waits for acknowledgment

3.4082

synchronous message communication with reply

tightly coupled message communication with reply

- 1.** form of communication in which a producer (or client) task sends a message to a consumer (or server) task and waits for a reply

3.4083

synchronous message communication without reply

tightly coupled message communication without reply

- 1.** a form of communication in which a producer task sends a message to a consumer task and waits for the consumer to accept the message

3.4084

synchronous request

- 1.** request where the client pauses to wait for completion of the request [ISO/IEC 19500-2:2012 *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.17]

3.4085

syntactic agreement

- 1.** passive interconnection in which two things agree on a set of symbols and symbol arrangements (statements) by which they will communicate [IEEE 1175.2-2006 *IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections*, 3.15]
cf. semantic agreement

3.4086

syntactic error

syntax error

- 1.** violation of the structural or grammatical rules defined for a language

cf. semantic error

EXAMPLE: in FORTRAN, using the statement $B + C = A$ rather than the correct $A = B + C$

3.4087

syntax

- 1.** structural or grammatical rules that define how the symbols in a language are to be combined to form words, phrases, expressions, and other allowable constructs **2.** structural components or features of a language and rules that define the ways in which the language constructs can be assembled together to form sentences [IEEE 1320.2-1998 (R2004) *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.195] **3.** definition of the format of information in a CDIF transfer [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2]
cf. semantics

3.4088

SYNTAX.1

- 1.** primary syntax defined within the CDIF family of standards [ISO/IEC 15474-1:2002 *Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: The CDIF family of standards supports multiple transfer formats, each composed of a syntax and an encoding.

3.4089

SyRS

- 1.** System Requirement Specification [ISO/IEC/IEEE 29148:2011 *Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

cf. SRS

3.4090

system

1. combination of interacting elements organized to achieve one or more stated purposes [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.38; ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.38; ISO/IEC TR 90005:2008 *Systems engineering — Guidelines for the application of ISO 9001 to system life cycle processes*, 2.1; ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.53; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.46] 2. product of an acquisition process that is delivered to the user [IEEE 15288.2:2014 *IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.1] 3. something of interest as a whole or as comprised of parts [ISO/IEC 10746-2:2009 *Information technology — Open Distributed Processing — Reference Model: Foundations*, 6.5] 4. interacting combination of elements to accomplish a defined objective [ISO/IEC TR 19759:2016 *Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOk)*, 1.1.6] 5. set of interrelated or interacting elements [ISO/IEC TR 90005:2008 *Systems engineering — Guidelines for the application of ISO 9001 to system life cycle processes*, 2.2]

Note 1 to entry: A system is sometimes considered as a product or as the services it provides. In practice, the interpretation of its meaning is frequently clarified by the use of an associative noun, e.g., aircraft system. Alternatively, the word 'system' can be replaced by a context-dependent synonym, e.g., aircraft, though this obscures the system perspective. A complete system includes all of the associated equipment, facilities, material, computer programs, firmware, technical documentation, services, and personnel required for operations and support to the degree necessary for self-sufficient use in its intended environment.

3.4091

system analysis

1. systematic investigation of a real or planned system to determine the information requirements and processes of the system and how these relate to each other and to any other system [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.4092

system breakdown structure (SBS)

1. system hierarchy, with identified enabling systems, and personnel that is typically used to assign development teams, support technical reviews, and to partition the assigned work and associated resource allocations to each of the tasks necessary to accomplish the technical objectives of the project [ISO/IEC/IEEE 24748-4:2016, *Systems and software engineering-Life cycle management-Part 4: Systems engineering planning*, 4.12]

Note 1 to entry: It also provides the basis for cost tracking and control.

3.4093

system description

1. documentation that results from system design defining the organization, essential characteristics and the hardware and software requirements of the system [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.4094

system design

1. process of defining the hardware and Software architecture, components, modules, interfaces and data for a system to satisfy specified requirements [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.4095

system design review (SDR)

1. review conducted to evaluate the manner in which the requirements for a system have been allocated to configuration items, the system engineering process that produced the allocation, the engineering planning for the next phase of the effort, manufacturing considerations, and the planning for production engineering
cf. critical design review, preliminary design review

3.4096

system development

1. process that usually includes requirements analysis, system design, implementation, documentation and quality assurance [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.4097

system development cycle

1. period of time that begins with the decision to develop a system and ends when the system is delivered to its end user

cf. system life cycle software development cycle

Note 1 to entry: This term is sometimes used to mean a longer period of time, either the period that ends when the system is no longer being enhanced, or the entire system life cycle.

3.4098

system documentation

1. collection of documents that describe the requirements, capabilities, limitations, design, operation, and maintenance of an information processing system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4099

system effectiveness analysis

1. analytical approach used to determine how well a system performs in its intended utilization environment [*IEEE 15288.1:2014 IEEE Standard for Application of Systems Engineering on Defense Programs, 3.1*]

3.4100

system element

1. member of a set of elements that constitutes a system [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.49; ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.22; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.54; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.47*]

cf. software/system element, software element

EXAMPLE: hardware, software, data, humans, processes (e.g. processes for providing service to users), procedures (e.g., operator instructions), facilities, materials, and naturally occurring entities (e.g., water, organisms, minerals), or any combination

Note 1 to entry: A system element is a discrete part of a system that can be implemented to fulfill specified requirements.

3.4101

system entity

1. in Mk II FPA, a contrivance which 'lumps together' all the non-primary entities of an application [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

3.4102

system follow-up

post-implementation review

post-development review

1. study of the effects of a system after it has reached a stabilized state of operational use [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4103

system hazard

1. system condition that is a prerequisite to an accident [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.8*]

cf. software hazard

3.4104

system integration

1. progressive assembling of system components into the whole system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4105

system integration module (SIM)

1. module in a microcontroller unit (MCU) that controls a system functional unit

3.4106

system interface task

1. task that hides the interface to and communicates with an external system or subsystem

3.4107

system library

1. software library containing system-resident software that can be accessed for use or incorporated into other programs by reference

cf. master library, production library, software development library, software repository

EXAMPLE: a macro library

3.4108

system life cycle

1. course of developmental changes through which a system passes from its conception to the termination of its use [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** period that begins when a system is conceived and ends when the system is no longer available for use

3.4109

system maintenance

1. modification of a system to correct faults, to improve performance, or to adapt the system to a changed environment or changed requirements [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4110

system model

1. in computer performance evaluation, a representation of a system depicting the relationships between workloads and performance measures in the system

cf. workload model

3.4111

system of systems (SoS)

1. system-of-interest (SOI) whose elements are themselves systems [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, G.1*] **2.** large system that delivers unique capabilities, formed by integrating independently useful systems

3.4112

system profile

1. set of measurements used in computer performance evaluation, describing the proportion of time each of the major resources in a computer system is busy, divided by the time that resource is available

3.4113

system requirements review (SRR)

1. review conducted to evaluate the completeness and adequacy of the requirements defined for a system; to evaluate the system engineering process that produced those requirements; to assess the results of system engineering studies; and to evaluate system engineering plans

cf. software requirements review

3.4114

system requirements specification (SyRS)

1. structured collection of information that embodies the requirements of the system **2.** structured collection of the requirements (functions, performance, design constraints, and attributes) of the system and its operational environments and external interfaces [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering, 4.1.29*]

cf. software requirements specification, SRS

3.4115

system safety

1. freedom from system hazards [*IEEE 1228-1994 (R2002) IEEE Standard for Software Safety Plans, 3.1.9*] *cf.* software safety

3.4116

system software

- 1.** software designed to facilitate the operation and maintenance of a computer system and its associated programs **2.** application-independent software that supports the running of application software [ISO/IEC 2382:2015, *Information technology — Vocabulary*] *cf.* application software, support software

EXAMPLE: operating systems, assemblers, utilities

3.4117

system specification

- 1.** documented set of mandatory requirements for a system [IEEE 15288.2:2014 *IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.1]

EXAMPLE: System Performance Specification (SPS), System Requirements Document (SRD), and System/Subsystem Specification (SSS).

3.4118

system structure

- 1.** decomposition of a system of interest into a set of interacting systems and system elements [ISO/IEC TR 29110-5-6-2:2014 *Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile*, 3.10]

Note 1 to entry: The system structure is described in a System Breakdown Structure (SBS).

3.4119

system support

- 1.** continued provision of services and material necessary for the use and improvement of an implemented system [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.4120

system table

- 1.** an entity type that cannot be maintained and, consequently, is not counted within the framework of FPA [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*]

3.4121

system test of user documentation

- 1.** testing performed with both the software and the documentation to evaluate that the documentation is fit for purpose and supports the users sufficiently in their use of the software [ISO/IEC 26513:2009 *Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.39]

3.4122

system testing

- 1.** testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements [IEEE 1012-2012 *IEEE Standard for System and Software Verification and Validation*, 3.1]

3.4123

system under test (SUT)

- 1.** parts of the CBSS to be tested [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.18]

Note 1 to entry: The SUT consists of hardware, system software, data communication features, or application software, or a combination of them.

3.4124

system-of-interest (SOI)

- 1.** system whose life cycle is under consideration [ISO/IEC 15026-3:2015 *Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.23; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.48] **2.** system whose life cycle is under consideration in the

application of verification and validation (V&V) [IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation]

3.4125

systematic failure

1. failure related in a deterministic way to a certain cause that can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation, or other relevant factors [ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.4.9]

3.4126

systematic reuse

1. practice of reuse according to a well-defined, repeatable process [IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3]

3.4127

systems engineering (SE)

1. interdisciplinary approach governing the total technical and managerial effort required to transform a set of stakeholder needs, expectations, and constraints into a solution, and to support that solution throughout its life [ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.6; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.56; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.49]

Note 1 to entry: System engineering is used when there is a single system-of-interest; systems engineering is for the discipline in general. It includes the definition of technical performance measures; the integration of engineering specialties toward the establishment of an architecture; and the definition of supporting lifecycle processes that balance cost, performance, and schedule objectives.

3.4128

Systems Engineering Plan (SEP)

Systems Engineering Management Plan (SEMP)

1. top level technical plan for managing the systems engineering effort which defines how the technical aspects of the project will be organized, structured, and conducted and how the systems engineering processes will be controlled to provide a product that satisfies stakeholder requirements [ISO/IEC/IEEE 24748-4:2016, Systems and software engineering—Life cycle management—Part 4: Systems engineering planning, 4.14]

2. top-level plan for managing the SE effort which, as such, defines how the project will be organized, structured, and conducted and how the total engineering process will be controlled to provide a product that satisfies stakeholder requirements [ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 3.5]

3. top-level technical planning document for a project which addresses technical management processes established by three principal sources: the project's contract or agreement, applicable organizational processes, and the systems engineering project team, as necessary to successfully accomplish the systems engineering-related tasks of the project [ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy, 4.60]

3.4129

systems integration testing

1. testing conducted on multiple complete, integrated systems to evaluate their ability to communicate successfully with each other and to meet the overall integrated systems' specified requirements.

3.4130

S_Packet

1. logically coherent grouping of STL sentences that describes a set of software concepts [IEEE 1175.3-2004 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior]

3.4131

T&E

1. test and evaluation [IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2]

3.4132

T&M

1. time and material [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4133

T-DUA

1. Trader Directory User Agent [*ISO/IEC 13235-3:1998 Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service, 4*]

3.4134

T-profile

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

3.4135

table

1. more concrete representation of an entity [*ISO/IEC 15476-4:2005 Information technology — CDIF semantic metamodel — Part 4: Data models, 6.3*]

3.4136

table heading

1. symbolic name or other means of referencing a decision table from other documents [*ISO 5806:1984 Information processing — Specification of single-hit decision tables, 3.12*]

Note 1 to entry: Alternatively, or in addition, a clear description of the table.

3.4137

table of contents

1. list of the headings in a document in order of appearance, with location indicators (such as page numbers) shown for each heading

3.4138

table-driven method

1. scheme that lets a program look up information in a table rather than using logic statements

3.4139

tacit knowledge

1. undocumented information [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4140

tag

1. symbolic name assigned to a specific release or a branch **2.** information structure that provides authoritative information about a software asset in order to facilitate its management

Note 1 to entry: provides developers and end users with a unique reference to the code base they are working with.

3.4141

tag creator

1. entity that initially creates a tag

Note 1 to entry: This entity can be part of the organization that created the software, in which case the tag creator and software creator will be the same. The tag creator can also be a third-party organization unrelated to the software creator, such as in the case where tags are created for legacy software by third-party organizations.

3.4142

tag slide

1. to apply the same tag to a changed version of a file to correct a last-minute error found in a release

3.4143

tagid

1. globally unique value that is globally unique for every SWID tag created [*ISO/IEC 19770-2:2015 Information technology — Software asset management — Part 2: Software identification tag, 3.1.3*]

3.4144

tailor

1. the act of carefully selecting process and related inputs and outputs contained within the PMBOK Guide® to determine a subset of specific processes that will be included within a project's overall management approach [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4145

tailored process

1. process developed by tailoring a standard process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.1.10]

3.4146

tailoring

1. adaptation of a software process by adding, modifying, and deleting process activities that are deemed inapplicable for the project [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4147

tailoring guideline

1. instructions that enable an organization to adapt standard processes appropriately to meet specific needs [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.1.11]

Note 1 to entry: Tailoring a process adapts the process description for a particular end. For example, a project creates its defined process by tailoring the organization's set of standard processes to meet the objectives, constraints, and environment of the project. The organization's set of standard processes is described at a general level that is not directly usable to perform a process. Tailoring guidelines aid those who establish the defined processes for specific needs. Tailoring guidelines describe what can and cannot be modified and identify process components that are candidates for modification.

3.4148

target entity

1. fundamental thing of relevance to the user, about which information is kept, and which needs to be measured [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*, 4.17]

Note 1 to entry: Possible synonyms of target entity are input to information product and work product. Examples of target entities are architecture, contextual schema, conceptual and logical and physical data models, data dictionary, document, data file, database management, relational database management system, form, and presentation device. Target entities are precisely defined by properties. Examples of properties are attribute, element, information, metadata, vocabulary, data format, data item, data value, information item, information item content, and data record.

3.4149

target language

object language

1. language in which the output from a machine-aided translation process is represented

cf. source language

EXAMPLE: the language output by an assembler or compiler

3.4150

target machine

1. computer on which a program is intended to execute **2.** computer being emulated by another computer
cf. host machine

3.4151

target node

1. node associated with the end of an arc [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.1.18]

3.4152

target of process

1. system, software product or task executed by system or software product to which measurement or evaluation process is applied [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.59]

3.4153

target process profile

1. process profile specifying which process attributes are required and the rating necessary for each process attribute for a required process [*ISO/IEC 33001:2015 Information technology — Process assessment — Concepts and terminology*, 3.2.20]

3.4154

target system

1. system to be categorized, which can be an IT system and software, including service provided by IT system [*ISO/IEC TR 12182:2015 Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it*, 3.4]

3.4155

task

1. required, recommended, or permissible action, intended to contribute to the achievement of one or more outcomes of a process [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.34; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.57; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.50] **2.** in software design, a software component that can operate in parallel with other software components **3.** activities required to achieve a goal [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information*, 2.13] **4.** concurrent object with its own thread of control **5.** sequence of instructions treated as a basic unit of work by the supervisory program of an operating system **6.** smallest unit of work subject to management accountability; a well-defined work assignment for one or more project members **7.** set or sequence of activities required to achieve a given goal [*ISO/IEC 25023:2016, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality*, 4.12]

Note 1 to entry: Related tasks are usually grouped to form activities.

3.4156

task behavior specification

1. specification describing a concurrent task's interface, structure, timing characteristics, relative priority, errors detected, and task event sequencing logic

3.4157

task completion

1. timely event when for a specific task the total output string or, in case of a set of output strings, all parts are completely received by the emulated user or another instance [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems*, 4.2]

Note 1 to entry: The time of task completion defines the end time of the preceding preparation time and the begin time of the execution time of the following task.

3.4158

task interface

1. input or output, events signaled (input or output), external inputs or outputs, or access to passive objects

3.4159

task inversion

1. optimization concept whereby the tasks in a system can be merged in a systematic way.

3.4160

task mode

1. indication of whether the user's preparation time begins immediately with the task submission of the preceding task (value = 0, i.e., "NO WAIT") or begins when the preceding task has been completed (task completion) (value = 1, i.e., "WAIT") [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems*, 4.21]

Note 1 to entry: mode of "Dialog" or "Batch" in UNIX-based systems.

3.4161

task priority criteria

- 1.** category of the task-structuring criteria addressing the relative importance of executing a given task

3.4162

task structuring

- 1.** software design stage with the objective of structuring a concurrent application into concurrent tasks and defining the task interfaces

3.4163

task submission

- 1.** timely event when the input string is completely submitted from the emulated user to the SUT and the execution of the task can start, regardless if the SUT starts the execution immediately or not [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems*, 4.22]

Note 1 to entry: Normally the task submission is defined internally by the submission of a special character (e.g. Carriage Return) or a character sequence at the end of the input string or at the end of several parts of the input string. Also it often happens that the task submission event is defined by the submission of the last character of any specified number characters in a string. For a classic batch task, the task submission is defined by the submission of the last character of the last string of the batch command sequence.

3.4164

task type

- 1.** classification of tasks which is defined by the combination of (1) the activity type, or a set of activity types which are all belonging to an identical timeliness function and task mode (2) the timeliness function; the task mode. [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems*, 4.23]

Note 1 to entry: Emulated users submit only these types of tasks to the SUT.

3.4165

task-clustering criteria

- 1.** category of the criteria addressing whether and how to group objects into concurrent tasks

3.4166

task-structuring criteria

- 1.** set of heuristics for helping a designer structure a system into concurrent tasks

3.4167

taxonomy

- 1.** scheme that partitions a body of knowledge and defines the relationships among the pieces **2.** classification scheme for referencing profiles or sets of profiles unambiguously [*ISO/IEC 29110-2-1:2015 — Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*, 4.63]

Note 1 to entry: It is used for classifying and understanding the body of knowledge.

3.4168

TBD

- 1.** to be determined [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

3.4169

TBR

- 1.** to be resolved [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2] **2.** to be revised [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

3.4170

TBS

- 1.** to be supplied [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.1] **2.** to be specified [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2]

3.4171

TCP

1. Transmission Control Protocol [*ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 4*]

3.4172

TCPI

1. to complete performance index

3.4173

TCS

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

3.4174

TCS-C

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

3.4175

TCS-W

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

3.4176

TDP

1. technical data package [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.4177

team selection plan

1. document specifying the qualifications, experience and training needs of project staff

3.4178

technical complexity adjustment

1. a factor which attempts to take into account the influence on application size of technical and quality requirements, which can be used to derive the adjusted size [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

Note 1 to entry: Note that if this is done, the result is not the functional size.

3.4179

technical complexity adjustment factors

1. the set of 19 factors that are taken into account in the technical complexity adjustment (TCA) [*ISO/IEC 20968:2002 Software engineering — Mk II Function Point Analysis — Counting Practices Manual, 10*]

Note 1 to entry: Each factor has a degree of influence (DI) between 1 and 5.

3.4180

technical contact

subject-matter expert

SME

1. person responsible for providing a documentation developer with technical information about a software product or for checking the technical accuracy of drafts of documentation [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.49*]

3.4181

technical debt

1. the deferred cost of work not done at an earlier point in the product life cycle [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4182

technical independence

1. of software quality assurance (SQA), situation in which the SQA effort uses personnel who are not involved in the development of the system or its elements [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes*, 3.2]

3.4183

technical management

1. application of technical and administrative resources to plan, organize, and control engineering functions [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.58]

3.4184

technical performance measure (TPM)

1. measure used to assess design progress, compliance to performance requirements, and technical risks for critical performance parameters [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 6.3.3.1]

3.4185

technical performance measurement

1. collection and comparison of technical accomplishments during project execution to the specified level of service, key performance indicators, or planned technical events and accomplishments

3.4186

Technical Report (TR)

1. document published by ISO or IEC containing collected data of a different kind from that normally published as an International Standard or Technical Specification [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.35]

EXAMPLE: data obtained from a survey carried out among the national bodies, data on work in other international organizations or data on the "state of the art" in relation to standards of national bodies on a particular subject

Note 1 to entry: [ISO/IEC Directives, Part 2]

3.4187

technical requirements

1. requirements relating to the technology and environment, for the development, maintenance, support and execution of the software

EXAMPLE: programming language, testing tools, operating systems, database technology and user interface technologies

3.4188

technical review

1. series of systems engineering activities conducted at logical transition points in a system life cycle, by which the progress of a program is assessed relative to its technical requirements using a mutually agreed-upon set of criteria [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs*, 3.1] **2.** systematic evaluation of a software product by a team of qualified personnel that examines the suitability of the software product for its intended use and identifies discrepancies from specifications and standards [*IEEE 1028-2008 IEEE Standard for Software Reviews and Audits*, 3.7]

Note 1 to entry: Technical reviews can also provide recommendations of alternatives and examination of various alternatives.

3.4189

technical standard

1. standard that describes the characteristics of applying accumulated technical or management skills and methods in the creation of a product or performing a service

3.4190

technique

1. a defined systematic procedure employed by a human resource to perform an activity to produce a product or result or deliver a service, and that may employ one or more tools [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **2.** methods and skills required to carry out a specific activity [*ISO/IEC 25001:2014 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Planning and management, 4.4*] **3.** technical or managerial procedure that aids in the evaluation and improvement of the software development process

3.4191

technology viewpoint

1. viewpoint on an ODP system and its environment that focuses on the choice of technology in that system [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 4.1.1.5*]

3.4192

TEMP

1. test and evaluation master plan [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.4193

template

<X> template

1. asset with parameters or slots that can be used to construct an instantiated asset [*IEEE 1517-2010 IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes, 3*] **2.** a partially complete document in a predefined format that provides a defined structure for collecting, organizing, and presenting information and data [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **3.** specification of the common features of a collection of <X>s in sufficient detail that an <X> can be instantiated using it [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.13*]

cf. construction

3.4194

template class

1. of an <X>, the set of all <X>s satisfying an <X> template type [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.23*]

3.4195

template type

1. of an <X>, a predicate defined in a template that holds for all the instantiations of the template and that expresses the requirements the instantiations of the template are intended to fulfill [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 9.22*]

3.4196

temporal clustering

1. task-structuring criterion by which activities that are not sequentially dependent, but are activated by the same event are grouped into a task

3.4197

temporal cohesion

1. type of cohesion in which the tasks performed by a software module are all required at a particular phase of program execution

cf. coincidental cohesion, communicational cohesion, functional cohesion, logical cohesion, procedural cohesion, sequential cohesion

EXAMPLE: a module containing all of a program's initialization tasks

3.4198

term

1. expression comprising constants, variables and operators built from a signature and a set of sorted variables [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and*

graphical notation, 2.1.24] 2. linguistic construct r to an entity [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 7.1]

3.4199

term evaluation

1. result obtained after the binding of variables in the term, the computation of the results of the associated functions, and any simplifications performed (such as gathering like terms to obtain the symbolic sum representation of a multiset) [ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.24.2]

3.4200

terminal

1. functional unit in a system or communication network at which data can be entered or retrieved [ISO/IEC 2382:2015, Information technology — Vocabulary]

3.4201

terminal symbol

1. part of the hierarchical definition of a syntax that is not further decomposed in the hierarchy [ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2]

3.4202

terminating behavior

1. behavior which breaks down a liaison and repudiates the corresponding contractual context and the corresponding contract [ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.2.5]

3.4203

termination construct

1. program construct that results in a halt or exit

3.4204

termination deliver

1. signal in the implicitly defined signal interface of a client computational object which has the same name and parameters as one of the terminations of an interrogation in the original operation interface [ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 3.3.13]
cf. invocation submit, invocation deliver

3.4205

termination submit

1. signal in the implicitly defined signal interface of a server computational object which has the same name and parameters as one of the terminations of an interrogation in the original operation interface [ISO/IEC 14752:2000 Information technology — Open Distributed Processing — Protocol support for computational interactions, 3.3.14]
cf. invocation submit, invocation deliver

3.4206

test

1. activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component [ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing, 4] **2.** to conduct an activity as in (1) **3.** set of one or more test cases and procedures.

3.4207

test approach

1. particular method that will be employed to pick the particular test case values.
cf. test practice

Note 1 to entry: varies in specificity from very general (e.g., black box or white box) to very specific (e.g., minimum and maximum boundary values)

3.4208

test basis

1. body of knowledge used as the basis for the design of tests and test cases [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.47]

Note 1 to entry: The test basis can take the form of documentation, such as a requirements specification, design specification, or module specification, but can also be an undocumented understanding of the required behavior.

3.4209

test bed

1. environment containing the hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test

3.4210

test case

1. set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.3.31] **2.** documentation specifying inputs, predicted results, and a set of execution conditions for a test item [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1.31] **2.** documentation specifying inputs, predicted results, and a set of execution conditions for a test item **3.** set of test case preconditions, inputs (including actions, where applicable), and expected results, developed to drive the execution of a test item to meet test objectives, including correct implementation, error identification, checking quality, and other valued information [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.48]

Note 1 to entry: A test case is the lowest level of test input (i.e. test cases are not made up of test cases) for the test subprocess for which it is intended

3.4211

test case generator

test data generator

test generator

1. software tool that accepts as input source code, test criteria, specifications, or data structure definitions; uses these inputs to generate test input data; and, sometimes, determines expected results

3.4212

test case specification

1. document specifying inputs, predicted results, and a set of execution conditions for a test item **2.** documentation of a set of one or more test cases [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*]

3.4213

test class

1. designated grouping of test cases.

3.4214

test completion process

1. test management process for ensuring that useful test assets are made available for later use, test environments are left in a satisfactory condition, and the results of testing are recorded and communicated to relevant stakeholders [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.50]

3.4215

test completion report

test summary report

1. report that summarizes the testing that was performed [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.51]

3.4216

test condition

1. testable aspect of a component or system, such as a function, transaction, feature, quality attribute, or structural element identified as a basis for testing [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.52]

3.4217

test coverage

1. degree, expressed as a percentage, to which specified test coverage items have been exercised by a test case or test cases [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.53] **2.** extent to which the test cases test the requirements for the system or software product [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.51]

3.4218

test coverage item

coverage item

1. attribute or combination of attributes that is derived from one or more test conditions by using a test design technique that enables the measurement of the thoroughness of the test execution [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.54]
cf. test item

3.4219

test criteria

1. criteria that a system or component must meet in order to pass a given test
cf. acceptance criteria, pass/fail criteria

3.4220

test data

1. data created or selected to satisfy the input requirements for executing one or more test cases, which can be defined in the test plan, test case, or test procedure [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes*, 4.34]

3.4221

test data readiness report

1. document describing the status of each test data requirement [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.56]

3.4222

test design

1. documentation specifying the details of the test approach for a system, software, or hardware feature or combination of features and identifying the associated tests [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1.32]

Note 1 to entry: commonly includes the organization of the tests into groups

3.4223

test design and implementation process

1. test process for deriving and specifying test cases and test procedures [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.57]

3.4224

test design specification

1. document specifying the features to be tested and their corresponding test conditions [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.58]

3.4225

test design technique

1. activities, concepts, processes, and patterns used to construct a test model that is used to identify test conditions for a test item, derive corresponding test coverage items, and subsequently derive or select test cases [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.59]

3.4226

test documentation

- 1.** documentation describing plans for, or results of, the testing of a system or component **2.** collection of the documentation inherent to the testing activities [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.19]

Note 1 to entry: Types include test case specification, test incident report, test log, test plan, test procedure, test report.

3.4227

test driver

- 1.** software module used to invoke a module under test and, often, provide test inputs, control and monitor execution, and report test results

cf. test harness

3.4228

test effort

- 1.** activity of performing one or more testing tasks.

3.4229

test environment

- 1.** facilities, hardware, software, firmware, procedures, and documentation intended for or used to perform testing of software [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.60]

Note 1 to entry: A test environment could contain multiple environments to accommodate specific test sub-processes, e.g., a unit test environment, a performance test environment.

3.4230

test environment readiness report

- 1.** document that describes the status of each environment requirement [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes*, 4.40]

3.4231

test environment requirements

- 1.** description of the necessary properties of the test environment [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.62]

3.4232

test environment set-up process

- 1.** dynamic test process for establishing and maintaining a required test environment [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.63]

3.4233

test execution

- 1.** process of running a test on the test item, producing actual results [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.64]

3.4234

test execution log

- 1.** document that records details of the execution of one or more test procedures [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.65]

3.4235

test execution process

- 1.** dynamic test process for executing test procedures created in the test design and implementation process in the prepared test environment, and recording the results [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.68]

3.4236

test harness

- 1.** scaffolding code written for the purpose of exercising lower level code when the higher-level code that will ultimately exercise it is not yet available
cf. test driver

3.4237

test incident report

- 1.** document reporting on any event that occurs during the testing process which requires investigation

3.4238

test incident reporting process

- 1.** dynamic test process for reporting to the relevant stakeholder's issues requiring further action that were identified during the test execution process [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.67]

3.4239

test item

test object

- 1.** work product that is an object of testing [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.68]

EXAMPLE: a system, a software item, a requirements document, a design specification, a user guide

3.4240

test item transmittal report

- 1.** document identifying test items

Note 1 to entry: contains current status and location information

3.4241

test level

test phase

- 1.** separate test effort that has its own documentation and resources **2.** specific instantiation of a test sub-process [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.69]

EXAMPLE: component, component integration, system, and acceptance testing

3.4242

test log

- 1.** chronological record of relevant details about the execution of tests

3.4243

test management

- 1.** planning, estimating, monitoring, reporting, control and completion of test activities [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes*, 4.49]

3.4244

test management process

- 1.** test process containing the sub-processes that are required for the management of a test project [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.71]
cf. test planning process, test monitoring and control process, test completion process

3.4245

test model

- 1.** representation of a test item that is used during the test case design process [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques*, 4.27]

3.4246

test monitoring and control process

1. test management process for ensuring that testing is performed in accordance with a test plan and with organizational test specifications [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.72]

3.4247

test objective

1. identified set of software characteristics to be measured under specified conditions by comparing actual behavior with the required behavior [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.20] 2. identified set of software features to be measured under specified conditions by comparing actual behavior with the required behavior [*ISO/IEC 25062:2006 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports*, 4.9]

3.4248

test phase

1. period of time in the software life cycle during which the components of a software product are evaluated and integrated, and the software product is evaluated to determine whether or not requirements have been satisfied 2. specific instantiation of test sub-process [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes*, 4.52]
cf. test level

3.4249

test plan

1. document that describes the technical and management approach to be followed to test a system or component [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1] 2. document describing the scope, approach, resources, and schedule of intended testing activities [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation; ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1; *ISO/IEC 25062:2006 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports*, 4.1] 3. plan that establishes detailed requirements, criteria, general methodology, responsibilities, and general planning for test and evaluation of a system [*ISO/IEC 2382:2015, Information technology — Vocabulary*] 4. detailed description of test objectives to be achieved and the means and schedule for achieving them, organized to coordinate testing activities for some test item or set of test items [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.75]

EXAMPLE: a project test plan (also known as a master test plan) that encompasses all testing activities on the project; further detail of particular test activities can be defined in one or more test sub-process plans (i.e. a system test plan or a performance test plan)

Note 1 to entry: It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. Typical contents identify the items to be tested, tasks to be performed, responsibilities, schedules, and required resources for the testing activity.

3.4250

test planning process

1. test management process used to complete test planning and develop test plans [*ISO/IEC TR 29119-1:2013 Software engineering—Software testing — Part 1: Concepts and definitions*, 4.76]

3.4251

test policy

organizational test policy

1. executive-level document that describes the purpose, goals, principles and scope of testing within an organization [*ISO/IEC/IEEE 29119-3:2013 Software and systems engineering — Software testing — Part 3: Test documentation*, 4.26]

3.4252

test practice

- 1.** conceptual framework that can be applied to the organizational test process, the test management process, or the dynamic test process to facilitate testing [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.77]
cf. test approach

3.4253

test procedure

- 1.** detailed instructions for the setup, execution, and evaluation of results for a given test case [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1.34] **2.** a document containing a set of associated instructions for testing [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1.34] **3.** documentation that specifies a sequence of actions for the execution of a test [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1.34] **4.** sequence of test cases in execution order, associated actions to set up the initial preconditions, and wrap-up activities post execution [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.78]

3.4254

test procedure specification

- 1.** document specifying a sequence of actions for the execution of a test **2.** document specifying one or more test procedures, which are collections of test cases to be executed for a particular objective [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.79]

Note 1 to entry: A test procedure specification for an automated test run is usually called a test script.

3.4255

test process

- 1.** process that provides information on the quality of a software product [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.80]

Note 1 to entry: often comprised of a number of activities, grouped into one or more test sub-processes

3.4256

test readiness review (TRR)

- 1.** review conducted to evaluate preliminary test results for one or more configuration items; to verify that the test procedures for each configuration item are complete, comply with test plans and descriptions, and satisfy test requirements; and to verify that a project is prepared to proceed to formal testing of the configuration items **2.** review as in (1) for any hardware or software component

cf. code review, formal qualification review, design review, requirements review

3.4257

test repeatability

- 1.** attribute of a test, indicating that the same results are produced each time the test is conducted

3.4258

test report

- 1.** document that describes the conduct and results of the testing carried out for a system or component
cf. test case specification, test completion report, test incident report, test item transmittal report, test log, test plan, test procedure

3.4259

test result

- 1.** indication of whether or not a specific test case has passed or failed, i.e. if the actual result observed as test item output corresponds to the expected result or if deviations were observed [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.82]

3.4260

test script

- 1.** test procedure specification for manual or automated testing [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.83*]

3.4261

test set

- 1.** set of one or more test cases with a common constraint on their execution [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.84*] **2.** collection of test cases for the purpose of testing a specific test objective [*ISO/IEC/IEEE 29119-2:2013 Software and systems engineering — Software testing — Part 2: Test processes, 4.62*]

Note 1 to entry: The test sets will typically reflect the feature sets, but they could contain test cases for a number of feature sets. Test cases for a test set could be selected based on the identified risks, test basis, retesting, or regression testing.

3.4262

test set architecture

- 1.** nested relationships between sets of test cases that directly reflect the hierachic decomposition of the test objectives.

3.4263

test specification

- 1.** complete documentation of the test design, test cases and test procedures for a specific test item [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.85*]

Note 1 to entry: A test specification could be detailed in one document, in a set of documents, or in other ways, for example, in a mixture of documents and database entries.

3.4264

test status report

- 1.** report that provides information about the status of the testing that is being performed in a specified reporting period [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.86*]

3.4265

test strategy

- 1.** part of the Test Plan that describes the approach to testing for a specific test project or test sub-process or sub-processes [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.87*]

Note 1 to entry: The test strategy usually describes some or all of the following: the test practices used; the test subprocesses to be implemented; the retesting and regression testing to be employed; the test design techniques and corresponding test completion criteria to be used; test data; test environment and testing tool requirements; and expectations for test deliverables.

3.4266

test sub-process

- 1.** test management and dynamic (and static) test processes used to perform a specific test level (e.g. system testing, acceptance testing) or test type (e.g. usability testing, performance testing) normally within the context of an overall test process for a test project [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.88*]

Note 1 to entry: Depending on the life cycle model used, test sub-processes are also typically called test phases, test levels, test stages or test tasks.

3.4267

test target version

- 1.** specific version of test target which is used for one-time execution of Dynamic Test Execution or Code Analysis [*ISO/IEC 30130:2016 Software engineering — Capabilities of software testing tools*]

3.4268

test traceability matrix

requirements test matrix

requirements verification table

verification cross reference matrix

1. document, spreadsheet, or other automated tool used to identify related items in documentation and software, such as requirements with associated tests [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.90]

cf. traceability matrix

3.4269

test type

1. group of testing activities that are focused on specific quality characteristics [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.91]

EXAMPLE: security testing, functional testing, usability testing, performance testing

3.4270

test unit

1. set of one or more computer program modules together with associated control data (for example, tables), usage procedures, and operating procedures that satisfy the following conditions: (a) All modules are from a single computer program; (b) At least one of the new or changed modules in the set has not completed the unit test; (c) The set of modules together with its associated data and procedures are the sole object of a testing process.

3.4271

testability

1. extent to which an objective and feasible test can be designed to determine whether a requirement is met [ISO/IEC 12207:2008 *Systems and software engineering — Software life cycle processes*, 4.52] **2.** degree of effectiveness and efficiency with which test criteria can be established for a system, product, or component and tests can be performed to determine whether those criteria have been met [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.7.5] **3.** degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met

3.4272

testing

1. activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component **2.** process of operating a system or component under specified conditions, observing or recording the results, and making an evaluation of some aspect of the system or component [ISO/IEC 25051:2014 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.23] **3.** set of activities conducted to facilitate discovery or evaluation of properties of one or more test items [ISO/IEC/IEEE 29119-1:2013 *Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.92]

3.4273

testing description

1. description of the test execution conditions (i.e. test procedure) [ISO/IEC 25051:2014 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.11; ISO/IEC 25062:2006 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports*, 4.11]

3.4274

testing task iteration

1. testing task that is re-performed during maintenance after having been originally performed during development

3.4275

testing tool

1. specific or generic tool which is used for test execution and test management such as test results recording, test results display, test results interpretation, generation of test data, generation of test procedure, generation of test scripts, test modelling [*ISO/IEC 30130:2016 Software engineering — Capabilities of software testing tools*]

3.4276

testware

1. products produced by the testing effort **2.** artifacts produced during the test process required to plan, design, and execute tests [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.93]

EXAMPLE: documentation, scripts, inputs, expected results, files, databases, environment, and any additional software or utilities used in the course of testing

3.4277

text

1. data in the form of characters, symbols, words, phrases, paragraphs, sentences, tables, or other character arrangements, intended to convey a meaning, and whose interpretation is essentially based upon the reader's knowledge of some natural language or artificial language [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: a screen. a business letter printed on paper or displayed on a screen

3.4278

text editor

editor

1. computer program, often part of a word processing system, that allows a user to enter, alter, and view text

3.4279

text page

1. model page that contains textual material related to a specific diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.102]

3.4280

text processing

word processing

1. data processing operations on text, such as entering, editing, merging, retrieving, storing, displaying, or printing [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4281

texture

architectural texture

1. collection of common development rules and constraints for realizing the applications of a product line [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering*, 3.19]

3.4282

theme

1. user stories associated by a common factor, such as functionality, data source, or security level [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4283

think time

1. elapsed time between the end of a prompt or message generated by an interactive system and the beginning of a human user's response

cf. port-to-port time, response time, turnaround time

3.4284

third normal form

1. result of a normalization process that transforms groups of data so that each non-key attribute does not depend on any other non-key attribute

3.4285

third party

1. person or body that is recognized as being independent of the parties involved, as concerns the issue in question [*ISO/IEC 25051:2014 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.25]

3.4286

thrashing

1. state in which a computer system is expending most or all of its resources on overhead operations, such as swapping data between main and auxiliary storage, rather than on intended computing functions

3.4287

thread

1. chain of actions, where at least one object participates in all the actions of the chain [*ISO/IEC 10746-1:1998 Information technology — Open Distributed Processing — Reference model: Overview*, 13.1.2]

3.4288

threat

1. state of the system or system environment which can lead to adverse effects **2.** a risk that would have a negative effect on one or more project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4289

threat agent

1. entity that can adversely act on property-of-interest [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*, 3.24]

3.4290

threat modeling

1. systematic exploration technique to expose any circumstance or event having the potential to cause harm to a system in the form of destruction, disclosure, modification of data, or denial of service.

Note 1 to entry: It results in a vulnerability assessment.

3.4291

three-address instruction

1. computer instruction that contains three address fields

cf. one-address instruction, two-address instruction, four-address instruction, zero-address instruction

EXAMPLE: an instruction to add the contents of locations A and B, and place the results in location C

3.4292

three-plus-one address instruction

1. computer instruction that contains four address fields, the fourth containing the address of the instruction to be executed next

cf. one-plus-one address instruction, two-plus-one address instruction, four-plus-one address instruction

EXAMPLE: an instruction to add the contents of locations A and B, place the results in location C, then execute the instruction at location D

Note 1 to entry: for example,

3.4293

three-point estimate

1. a technique used to estimate cost or duration by applying an average of optimistic, pessimistic, and most likely estimates when there is uncertainty with the individual activity estimates [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4294

threshold

1. a cost, time, quality, technical, or resource value used as a parameter, and which may be included in product specifications. Crossing the threshold should trigger some action, such as generating an exception report. [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4295

throughput

1. amount of work that can be performed by a computer system or component in a given period of time **2.** rate (i.e., the average number per time unit with respect to the rating interval) of all tasks of a task type submitted to the SUT [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.24*]

Note 1 to entry: Usually throughput is defined by the rate of terminated tasks during a period of time.

3.4296

throughput rating value

1. quotient (corresponding to the j-th task type) of the (actual) throughput and the throughput reference value [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.25*]

3.4297

throughput reference value

1. minimum throughput required by the set of emulated users [*ISO/IEC 14756:1999 Information technology — Measurement and rating of performance of computer-based software systems, 4.26*]

3.4298

thumbnail

1. miniature image file displayed for quick identification of a larger image or video file [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.24*]

3.4299

tier

1. grouping of process definitions [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.50*]

3.4300

TIM

1. technical interchange meeting [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.4301

time

1. in decreasing order of resolution, CPU execution time, elapsed time (i.e., wall clock time), or calendar time [*IEEE 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability, 2.6*]

3.4302

time and material (T&M) contract

1. a type of contract that is a hybrid contractual arrangement containing aspects of both cost-reimbursable and fixed-price contracts. Time and material contracts resemble cost-reimbursable type arrangements in that they have no definitive end, because the full value of the arrangement is not defined at the time of the award. Thus, time and material contracts can grow in contract value as if they were cost-reimbursable-type arrangements. Conversely, time and material arrangements can also resemble fixed-price arrangements. For example, the unit

rates are preset by the buyer and seller, when both parties agree on the rates for the category of senior engineers
[A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4303

time behavior

1. degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.2.1]

3.4304

time class

1. time limit, combined with a relative frequency corresponding to the ratio of the number of tasks (of a specific task type) with an execution time less than or equal to the corresponding time limit, to the total number of tasks (of that particular task type), used for comparison with the execution time of a task (of that particular task type) [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.27]

3.4305

time out

1. condition that occurs when a predetermined amount of time elapses without the occurrence of an expected event**2.** to experience the condition in (1)

EXAMPLE: the condition that causes termination of an online process if no user input is received within a specified period of time

3.4306

time sharing

1. mode of operation that permits two or more users to execute computer programs concurrently on the same computer system by interleaving the execution of their program

Note 1 to entry: Time sharing can be implemented by time slicing, priority-based interrupts, or other scheduling methods.

3.4307

time slicing

1. mode of operation in which two or more processes are each assigned a small, fixed amount of continuous processing time on the same processor, and the processes execute in a round-robin manner, each for its allotted time, until all are completed

3.4308

time-boxed

1. having a prescribed duration limit for a project task [Software Extension to the PMBOK® Guide Fifth Edition]

3.4309

time-critical task

1. task that must meet a hard deadline

3.4310

time-scaled schedule network diagram

1. any project schedule network diagram drawn in such a way that the positioning and length of the schedule activity represents its duration. Essentially, it is a bar chart that includes schedule network logic [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4311

timeliness function

1. description of the user requirements with respect to the execution times of tasks of a specific task type [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.29]

Note 1 to entry: It consists of one or more time classes.

3.4312

timeliness rating value

1. quotient (corresponding to the j-th task type) of the timely throughput and the total throughput [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.28]

3.4313

timely throughput

1. throughput of all of those tasks whose execution times are accepted with respect to the timeliness function [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.3]

3.4314

timer event

1. stimulus used to periodically activate a task

3.4315

timer pulse unit (TPU)

1. microcontroller unit to generate timed pulses

3.4316

timesharing

1. operating technique of a data processing system that provides for the interleaving in time of two or more processes in one processor [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

3.4317

timing

1. process of estimating or measuring the amount of execution time required for a software system or component
cf. sizing

3.4318

timing analysis

1. method to evaluate the time sequence of logic signals, or the speed of a digital circuit

3.4319

timing analyzer

1. software tool that estimates or measures the execution time of a computer program or portion of a computer program, either by summing the execution times of the instructions along specified paths or by inserting probes at specified points in the program and measuring the execution time between probes

3.4320

timing diagram

1. diagram showing the time-ordered execution sequence of a group of tasks

3.4321

TINA

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

3.4322

tinderbox

1. automated build and regression-testing tool

Note 1 to entry: A tinderbox will typically fetch on a regular basis the latest versions of the software from each supported branch, build it for the different platforms, and report the results from the build and the regression tests.

3.4323

TLS

1. Transport Layer Security [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.4324

TMRR

1. technology maturation and risk reduction [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.4325

to-complete performance index (TCPI)

1. a measure of the cost performance that must be achieved with the remaining resources in order to meet a specified management goal, expressed as the ratio of the cost to finish the outstanding work to the remaining budget [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4326

TOC

1. total ownership cost [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]

3.4327

token

1. data item associated with a place and chosen from the place's type [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation, 2.1.25*] **2.** terminal symbol [*ISO/IEC 15475-2:2002 Information technology — CDIF transfer format — Part 2: Syntax SYNTAX.1, 6.1*]

3.4328

tolerable risk

1. level of risk that is accepted in a given context based on the current values of society [*ISO/IEC 15026-3:2015 Systems and software engineering — Systems and software assurance — Part 3: System integrity levels, 3.25*]

3.4329

tolerance

1. the quantified description of acceptable variation for a quality requirement [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4330

tool

1. software product that provides support for software and system life cycle processes [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*] **2.** something tangible, such as a template or software program, used in performing an activity to produce a product or result [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] **3.** device that performs or assists in the performance of user or organization process tasks that support, directly or indirectly, the achievement of production goals [*IEEE 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections, 3.16*]

Note 1 to entry: particularly, but not exclusively, a modeling tool. Also, tool is used as a short form for software tool, and more specifically for CASE tool.

3.4331

tool-specific information

1. information associated with an object of a net graph or with the net graph itself that is specific to a particular tool and is not meant to be used by other tools [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format, 4.1.19*]

3.4332

top box

1. box in the A-0 context diagram that models the top-level function of an IDEF0 model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.103*]

3.4333

top-down

1. pertaining to an activity that starts with the highest-level component of a hierarchy and proceeds through progressively lower-levels **2.** pertaining to a method or procedure that starts at the highest level of abstraction and proceeds towards the lowest level [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. bottom-up, critical piece first

EXAMPLE: top-down design, top-down testing

3.4334

top-down design

top-down decomposition

- 1.** design approach in which a system's functionality is decomposed from high-level concepts into lower-level pieces
- 2.** process of designing a system by identifying its major components, decomposing them into their low-level components, and iterating until the desired level of detail is achieved

3.4335

top-level function

- 1.** function modeled by the single box in the A-0 context diagram of an IDEF0 model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.104]

3.4336

topic

- 1.** small part of a document that deals with a single subject [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.50]

EXAMPLE: instructions on how to print the current document

Note 1 to entry: In printed documentation, a topic is equivalent to a section (heading; subheading) and its content. In onscreen documentation, a topic consists of a title (heading) and information about a subject (typically, a task or a concept or reference information). For on-screen documentation, the system can present a topic without user intervention.

3.4337

tornado diagram

- 1.** a special type of bar chart used in sensitivity analysis for comparing the relative importance of the variables [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4338

total

- 1.** complete mapping [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.196]

cf. partial, mandatory, mapping completeness

Note 1 to entry: The mapping M from a set D to a set R is total if for every X in D, there is at least one Y in R and pair [X,Y] in M. A property of a class is total, meaning that it will have a value for every instance of the class, unless it is explicitly declared partial.

3.4339

total cluster

complete cluster

- 1.** subclass cluster in which each instance of a superclass must be an instance of at least one of the subclasses of the cluster [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.197]

cf. incomplete cluster, partial cluster, superclass

3.4340

total correctness

- 1.** in proof of correctness, a designation indicating that a program's output assertions follow logically from its input assertions and processing steps

cf. partial correctness

3.4341

total float (TF)

- 1.** the amount of time that a schedule activity can be delayed or extended from its early start date without delaying the project finish date or violating a schedule constraint [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4342

total quality management (TQM)

1. holistic approach to quality improvement in all life-cycle phases

3.4343

TP

1. Transaction Processing [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 4]

3.4344

TPM

1. technical performance measure [*ISO/IEC/IEEE 29148:2011 Systems and software engineering — Life cycle processes — Requirements engineering*, 4.2] **2.** technical performance management [*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]

3.4345

TPU

1. timer pulse unit

3.4346

TQM

1. Total Quality Management

3.4347

TR

1. technical requirements [*ISO/IEC TR 14143-4:2002 Information technology — Software measurement — Functional size measurement — Part 4: Reference model*, 4] **2.** ODP Type Repository [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function*, 4]

3.4348

TRA

1. threat and risk assessment [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]

3.4349

trace

1. record of the execution of a computer program, showing the sequence of instructions executed, the names and values of variables, or both **2.** to produce a record as in (1) **3.** to establish a relationship between two or more products of the development process **4.** record of an object's interactions, from its initial state to some other state [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.7]

Note 1 to entry: Types include execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace.

3.4350

traceability

1. degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4] **2.** discernible association among two or more logical entities, such as requirements, system elements, verifications, or tasks [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.71] **3.** degree to which each element in a software development product establishes its reason for existing [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities*, 3.14]

EXAMPLE: the degree to which the requirements and design of a given system element match, the degree to which each element in a bubble chart references the requirement that it satisfies

3.4351

traceability matrix

- 1.** matrix that records the relationship between two or more products of the development process

EXAMPLE: a matrix that records the relationship between the requirements and the design of a given software component

3.4352

traceable

- 1.** having components whose origin can be determined [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation)*, 5.25]

3.4353

trade secret

- 1.** formula, process, design, or intellectual property that is protected by non-disclosure

3.4354

trade study

- 1.** evaluation of alternatives, based on criteria and systematic analysis, to select the best alternative for attaining determined objectives

3.4355

trade-off

- 1.** decision-making actions that select from various requirements and alternative solutions on the basis of net benefit to the stakeholders [*ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.59; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.51]

3.4356

trade-off analysis

- 1.** analytical evaluation of design options/alternatives against performance, design-to-cost objectives, and life cycle quality factors

3.4357

trademark

- 1.** symbol, word, or phrase used to denote a particular source of goods or services

3.4358

trading

- 1.** interaction between objects in which information about new or potential contracts is exchanged via a third-party object [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 13.5.5]

3.4359

trailer

- 1.** Identification or control information placed at the end of a file or message
cf. header (2)

3.4360

trailing decision

- 1.** loop control that is executed after the loop body
cf. leading decision UNTIL

3.4361

training

- 1.** provision of formal and informal learning activities

Note 1 to entry: includes in-class instruction, informal mentoring, Web-based tutorials, guided self-study, and formalized on-the-job exercises. The learning options selected for each situation are based on an assessment of the performance gap to be addressed and resources.

3.4362

transaction

1. in software engineering, a data element, control element, signal, event, or change of state that causes, triggers, or initiates an action or sequence of actions **2.** activity which leads to a set of object changes consistent with a dynamic schema (and its constraining invariant schema) [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 13.7.1.1*]

3.4363

transaction analysis

transaction-centered design

1. software development technique in which the structure of a system is derived from analyzing the transactions that the system is required to process

cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transform analysis

3.4364

transaction file

1. a temporary data file [*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*]

Note 1 to entry: it is read one time only and its data is consumed.

3.4365

transaction matrix

1. matrix that identifies possible requests for database access and relates each request to information categories or elements in the database

3.4366

transaction schema

1. dynamic schema and an invariant schema defining transactions and their dependencies [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 16.8.1.1*]

3.4367

transaction transparency

1. distribution transparency which masks coordination of activities amongst a configuration of objects to achieve consistency [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture, 4.4.1.8*]

3.4368

transactional function

1. elementary process that provides functionality to the user to process data [*ISO/IEC 20926:2009 Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009, 3.49*] **2.** a transaction [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

Note 1 to entry: A succession of actions which the user sees as a single work unit. FPA assigns each transactional function a type and therefore distinguishes between the following types: external input, external output, and external inquiry.

3.4369

transactional function type

1. one of three categories that FPA assigns to a transactional function external input, external output, and external inquiry [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.4370

transfer

1. to send data from one place and receive it at another **2.** to relinquish control by one process and assume it at another, either with or without expectation of return

cf. jump

3.4371

transfer file

- 1.** file containing data to be interchanged [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.1]

Note 1 to entry: It is made up of a header and a number of components. Components contain either data or data definition data.

3.4372

transfer of an entitlement

- 1.** process of assigning a given entitlement to a separate legal entity [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.41]

Note 1 to entry: Transfers often occur when a large organization divests a part of itself into a separate legal entity. The Ent enables the recording of entitlement transfers. Transfers are in accordance with the contractual terms and conditions between the software licensor and end-user.

3.4373

transform analysis

transformation analysis

transform-centered design

- 1.** software development technique in which the structure of a system is derived from analyzing the flow of data through the system and the transformations that must be performed on the data
cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transaction analysis

3.4374

transient error

- 1.** error that occurs once, or at unpredictable intervals

cf. intermittent fault, random failure

3.4375

transition

- 1.** activities involved in moving a new or changed service, system, or component to or from an environment **2.** change from one state to another state or the same state [*ISO/IEC 11411:1995 Information technology — Representation for human communication of state transition of software*, 2.2] **3.** node of a net, taken from the transition kind, and represented by a rectangle in the net graph [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26]

3.4376

transition condition

- 1.** Boolean expression (one that evaluates to true or false) associated with a transition [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26.1]

3.4377

transition mode

- 1.** pair comprising the transition and a mode [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26.2]

3.4378

transition occurrence

transition rule

- 1.** if a transition is enabled in a mode, it can occur in that mode. [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26.3]

Note 1 to entry: On the occurrence of the transition, the following actions occur indivisibly 1. for each input place of the transition the enabling tokens of the input arc with respect to that mode are subtracted from the input place's marking, and 2. for each output place of the transition the multiset of tokens of the evaluated output arc expression is added to the marking of the output place. A place can be both an input place and an output place of the same transition.

3.4379

transition variables

1. variables that occur in the expressions associated with the transition [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.26.5]

Note 1 to entry: These are the transition condition and the annotations of arcs surrounding the transition.

3.4380

translator

1. computer program that transforms a sequence of statements expressed in one language into an equivalent sequence of statements expressed in another language

cf. assembler, compiler

3.4381

trap

1. conditional jump to an exception or interrupt handling routine, often automatically activated by hardware, with the location from which the jump occurred recorded **2.** to perform the operation in (1)

3.4382

tree diagram

1. a systematic diagram of a decomposition hierarchy used to visualize as parent-to-child relationships a systematic set of rules [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4383

tree-structured chart

1. chart depicting program constructs defined in ISO/IEC 8631 and having the structure of a tree [*ISO/IEC 14568:1997 Information technology — DXL: Diagram eXchange Language for tree-structured charts*, 3.1.1]

3.4384

trend analysis

1. an analytical technique that uses mathematical models to forecast future outcomes based on historical results. It is a method of determining the variance from a baseline of a budget, cost, schedule, or scope parameter by using prior progress reporting periods' data and projecting how much that parameter's variance from baseline might be at some future point in the project if no changes are made in executing the project [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4385

trigger condition

1. an event or situation that indicates that a risk is about to occur [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4386

triggering event

triggering event type

1. event (something that happens) that causes a functional user of the piece of software to initiate (trigger) one or more functional processes [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method*, 2.25]

3.4387

triple constraint

1. framework for evaluating competing demands, such as schedule, cost, and quality

Note 1 to entry: The triple constraint is often depicted as a triangle where one of the sides or one of the corners represents one of the parameters being managed by the project team.

3.4388

TRR

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

3.4389

trunk

- 1.** software's main line of development; the main starting point of most branches

Note 1 to entry: One can often distinguish the trunk from other branches by the version numbers used for identifying its files, which are shorter than those of all other branches.

3.4390

trust

- 1.** degree to which a user or other stakeholder has confidence that a product or system will behave as intended
[*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.3.2]

3.4391

tunnel notation

- 1.** pair of short shallow arcs, resembling a pair of left and right parentheses characters, that bracket the arrowhead or the arrowtail of an arrow segment [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.107]

3.4392

tunneled arrow

- 1.** arrow left undrawn between its attachment to an ancestral box and its appearance as a boundary arrow on some hierarchically consecutive descendent diagram [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.105]

3.4393

tunneling

- 1.** act of applying tunnel notation to an arrow segment [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*, 2.1.106]

3.4394

tuple

- 1.** set of fields or data items [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.37]

Note 1 to entry: Tuple can be used in place of record.

3.4395

turnaround time

- 1.** elapsed time between the submission of a job to a batch processing system and the return of completed output
cf. port-to-port time, response time, think time

3.4396

turnkey

- 1.** pertaining to a hardware or software system delivered in a complete, operational state

3.4397

turnkey system

- 1.** data processing system that is ready to use when installed and supplied to the user in a ready-to-run condition, possibly customized to a specific user or application [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

Note 1 to entry: Some preparatory work on the user's data can be required.

3.4398

tutorial

- 1.** instructional procedure in which the user exercises software functions using sample data that is supplied with the software or documentation [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.51]

3.4399

two-address instruction

double-operand instruction

1. computer instruction that contains two address fields.

cf. one-address instruction, three-address instruction, four-address instruction, zero-address instruction

EXAMPLE: an instruction to add the contents of A to the contents of B

3.4400

two-level address

1. indirect address that specifies the storage location containing the address of the desired operand
cf. n-level address

3.4401

two-level encoding

1. microprogramming technique in which different microoperations can be encoded identically into the same field of a microinstruction, and the one that is executed depends upon the value in another field internal or external to the microinstruction.

cf. bit steering, residual control, single-level encoding

3.4402

two-phase acquisition

1. segmenting a project into an early phase that focuses on gathering requirements, addressing major risks, and project planning; and a later phase that completes the project if the outcome of the first phase is favorable.

Note 1 to entry: The final decision on whether to do the full project is deferred from the point when the uncertainties are the greatest (the beginning) to a point where the uncertainties are significantly reduced.

3.4403

two-plus-one address instruction

1. computer instruction that contains three address fields, the third containing the address of the instruction to be executed next

cf. one-plus-one address instruction, three-plus-one address instruction, four-plus-one address instruction

EXAMPLE: an instruction to add the contents of A to the contents of B, then execute the instruction at location C

3.4404

type

1. set [*ISO/IEC 15909-1:2004 Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*, 2.1.27] **2.** of an <X>, a predicate characterizing a collection of <X>s [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations*, 9.9] **3.** identifiable entity with an associated predicate (a single-argument mathematical function with a Boolean result) defined over entities [*ISO/IEC 19500-1:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*, 5.3.4]
cf. class

3.4405

UART

1. universal asynchronous receiver/transmitter

3.4406

UDF

1. unit development folder

cf. software development file

3.4407

ULA

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

3.4408

UML

1. Unified Modeling Language [*ISO/IEC 14769:2001 Information technology — Open Distributed Processing — Type Repository Function, 4; ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications, 4*]

3.4409

unambiguous

1. described in terms that only allow a single interpretation, aided, if necessary, by a definition [*ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.26*]

3.4410

unanimity

1. agreement by everyone in the group on a single course of action [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4411

unbind behavior

1. behavior that terminates a binding [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 13.5.4*]

3.4412

unbundle

1. separation of arrow meanings, expressed by branching arrow segments [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.108*]

Note 1 to entry: That is, the separation of object types from an object type set.

3.4413

uncertainty

1. result of not having accurate or sufficient knowledge of a situation **2.** state, even partial, of deficiency of information related to understanding or knowledge of an event, its consequence, or likelihood [*ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.40*]

cf. risk

Note 1 to entry: often the root cause of a risk factor. In business decision making, uncertainty refers to unquantified variation; the probabilities of the variations cannot be used in the decision analysis.

3.4414

unconditional jump

1. jump that takes place regardless of execution conditions

cf. conditional jump

3.4415

underflow exception

1. exception that occurs when the result of an arithmetic operation is too small a fraction to be represented by the storage location designated to receive it

cf. addressing exception, data exception, operation exception, overflow exception, protection exception

3.4416

underlying license

1. license for software use as originally purchased or procured, and which can typically be linked directly to purchase records [*ISO/IEC 19770-1:2012 Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance, 3. 15*]

Note 1 to entry: An underlying license can have conditions associated with it, requiring it to be used in combination with another license or licenses to create an effective full license. It can also have capacity or permission to use future versions of the software, or specify ways or limitations to how it can be upgraded or replaced by a new version, or how the license can be upgraded by combining with another license that is linked directly to another purchase record.

3.4417

understandability

- 1.** ease with which a system can be comprehended at both the system-organizational and detailed-statement levels

Note 1 to entry: Understandability has to do with the system's coherence at a more general level than readability does.

3.4418

undirected graph

- 1.** graph (sense 2) in which no direction is implied in the internode connections
cf. directed graph

3.4419

unidimensionality

- 1.** existence of a single trait or construct underlying a set of measures [*ISO/IEC 33003:2015 Information technology — Process assessment — Requirements for process measurement frameworks*, 3.15]

3.4420

Unified Modeling Language (UML)

- 1.** graphical language for visualizing, specifying, constructing, and documenting an object-oriented software-intensive system's artifacts

3.4421

uniform resource identifier (URI)

- 1.** compact sequence of characters that identifies an abstract or physical resource available on the Internet [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.51]

Note 1 to entry: The syntax used for URIs is defined in IETF RFC 3986.

3.4422

Uniform Resource Locator (URL)

- 1.** mechanism for identifying resources on the Internet (such as web pages) by specifying the address of the resource and the access protocol used [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.25]

Note 1 to entry: The term as specified by the IETF is uniform resource identifier (URI) of which URL is a subset.

3.4423

unique function

- 1.** a function that differs in form and/or logical processing from every other function provided by a certain application [*ISO/IEC 24570:2005 Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

3.4424

uniqueness constraint

- 1.** constraint stating that no two distinct instances of a class agree on the values of all the properties that are named in the uniqueness constraint [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.199]

3.4425

unit

- 1.** separately testable element specified in the design of a computer software component **2.** logically separable part of a computer program **3.** software component that is not subdivided into other components **4.** distinguishable architectural unit with individual identity, boundary, and behavior that is observable through interactions with other such units [*IEEE 1175.4-2008 IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*] **5.** piece or complex of apparatus serving to perform one particular function [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4] **6.** software element that is not subdivided into other elements [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4]

3.4426

unit (of measure)

- 1.** particular quantity defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitude relative to that quantity [*ISO/IEC 25021:2012 Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements, 4.18*]
cf. unit of measurement

Note 1 to entry: Only quantities expressed in the same units of measurement are directly comparable. Examples of units include the number of faults and the number of failures. Hour and meter are also units of measure. Units of measurement have conventionally assigned names and symbols.

3.4427

unit of measurement

- 1.** particular quantity defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitudes relative to that quantity [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process, 3.39; ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.26*]
cf. unit (of measure)

Note 1 to entry: Units of measurement have conventionally assigned names and symbols.

3.4428

unit requirements documentation

- 1.** documentation that sets forth the functional, interface, performance, and design constraint requirements for the test unit.

3.4429

unit test

- 1.** testing of individual routines and modules by the developer or an independent tester **2.** test of individual programs or modules in order to ensure that there are no analysis or programming errors [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **3.** test of individual hardware or software units or groups of related units

3.4430

unit test framework

- 1.** environment that facilitates unit testing

3.4431

universal asynchronous receiver/transmitter (UART)

- 1.** electronic unit used for serial communications that translates data between parallel and serial forms

Note 1 to entry: The data format and transmission speeds are configurable. Commonly part of a microcontroller.

3.4432

universal serial bus (USB)

- 1.** serial communication interface with two data lines and two power lines between a computer and peripherals

3.4433

unpack

- 1.** to recover the original form of one or more data items from packed data
cf. pack

3.4434

unscripted testing

- 1.** dynamic testing in which the tester's actions are not prescribed by written instructions in a test case [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions, 4.94*]

3.4435

UNSP-SC

- 1.** United Nations standard products and services code [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema, 3.2*]

3.4436

unspecialize

1. change by an instance from being an instance of its current subclass within a cluster to being an instance of none of the subclasses in the cluster [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.200*]
cf. respecialize, specialize

3.4437

unstratified language

1. language that can be used as its own metalanguage

EXAMPLE: English, German

3.4438

UNTIL

post-tested iteration

1. single-entry, single-exit loop, in which the loop control is executed after the loop body.
cf. closed loop, WHILE, trailing decision

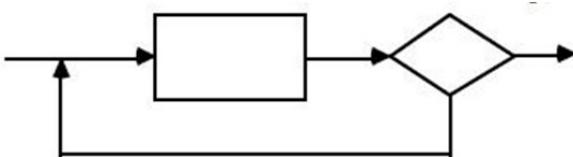


Figure 18 — UNTIL construct

3.4439

unwind

1. in programming, to state explicitly and in full all of the instructions involved in multiple executions of a loop
cf. straight-line coding

3.4440

UOD

1. Universe of Discourse [*ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications, 4*]

3.4441

up

1. pertaining to a system or component that is operational and in service
cf. busy, down, idle

Note 1 to entry: Such a system is either busy or idle.

3.4442

up time

1. period of time during which a system or component is operational and in service; that is, the sum of busy time and idle time

cf. down time, busy time, idle time, mean time between failures, set-up time

3.4443

updatable argument

1. designation given to an operation argument that identifies an instance to which a request can be sent that will change the state of the instance [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.201*]

Note 1 to entry: An argument not designated as "updatable" means that no requests can be sent that can change the state of the instance identified by the argument.

3.4444

upload

- 1.** to transfer programs or data from a connected computer to a computer with greater resources [ISO/IEC 2382:2015, *Information technology — Vocabulary*]

Note 1 to entry: typically, from a personal computer to a server

3.4445

upward compatible

- 1.** pertaining to hardware or software that is compatible with a later or more complex version of itself
cf. downward compatible

EXAMPLE: a program that handles files created by a later version of itself

3.4446

upward compression

- 1.** in software design, a form of demodularization in which a subordinate module is copied inline into the body of a superordinate module
cf. lateral compression, downward compression

3.4447

UR

- 1.** user requirements [ISO/IEC TR 14143-4:2002 *Information technology — Software measurement — Functional size measurement — Part 4: Reference model*, 4]

3.4448

URa

- 1.** utilize benchmarking results activity [ISO/IEC 29155-2:2013: *Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*, 4]

3.4449

URI

- 1.** uniform resource identifier [ISO/IEC 19770-5:2015 *Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.51]

3.4450

URL

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

3.4451

usability

- 1.** extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use [ISO/IEC 25064:2013 *Systems and software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: User needs report*, 4.16] **2.** degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.4]
cf. reusability

Note 1 to entry: Usability can either be specified or measured as a product quality characteristic in terms of its subcharacteristics, or specified or measured directly by measures that are a subset of quality in use.

3.4452

usability defect

- 1.** product attribute that leads to a mismatch between user intentions or user actions and the system attributes and behavior [ISO/IEC 25066:2016, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report*, 3.17]

EXAMPLE: additional unnecessary steps not required as part of completing a task, misleading information, insufficient or poor information on the user interface, unexpected system responses, limitations in navigation, inefficient use error recovery mechanisms, physical characteristics of the user interface that are not suitable for the physical characteristics of the user, deviations of product attributes from established criteria

3.4453

usability finding

1. identified usability defect, usability problem, or positive usability-related attribute [*ISO/IEC 25066:2016, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report, 3.18*]

3.4454

usability inspection

1. evaluation based on the considered judgment of evaluators who examine the usability-related aspects of an interface with respect to specified criteria [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.16*]

Note 1 to entry: Usability inspection is the generic term for several methods, including but not limited to heuristic evaluation, cognitive walkthroughs, standards inspection, pluralistic walkthroughs, and consistency inspections. The inspectors can include usability specialists, developers, end users or other types of professionals. The evaluative criteria can include good practice and/or documented principles, guidelines, requirements or standards. The evaluation can be conducted with or without the help of referenced documents.

3.4455

usability laboratory

1. typically, a suite of evaluation and observation rooms fitted with video and audio equipment for recording user responses

3.4456

usability objective

1. stated level of usability expressed in terms of effectiveness, efficiency and satisfaction in a specified context of use which can be verified [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.15*]

Note 1 to entry: Usability objectives can be stated as user requirements, in which case the level to be achieved is a usability requirement, or they can be stated as desired target levels, depending on their use in design and evaluation.

3.4457

usability problem

1. situation during use resulting in poor effectiveness, efficiency or satisfaction [*ISO/IEC 25066:2016, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report, 3.19*]

3.4458

usability test

fitness-for-use test

1. test to determine whether an implemented system fulfils its functional purpose as determined by its users [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. usability testing

3.4459

usability walkthrough

1. usability evaluation in which one or more evaluators step through a scenario playing the role of a user and identifying usability problems associated with successful completion of the scenario [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.18*]

Note 1 to entry: The evaluators can include usability specialists, developers, end users, or other types of professionals.

3.4460

usage mode

1. primary manner in which the documentation developer expects the document to be used [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.53]

3.4461

USB

1. universal serial bus

3.4462

use case

1. in UML, a complete task of a system that provides a measurable result of value for an actor **2.** sequence of tasks that a system can perform, interacting with users of the system and providing a measurable result of value for the user [*ISO/IEC 26513:2009 Systems and software engineering — Requirements for testers and reviewers of user documentation*, 3.43] **3.** description of the behavioral requirements of a system and its interaction with a user [*ISO/IEC/IEEE 26515: 2011 Systems and software engineering: Developing user documentation in an agile environment*, 4.15]

Note 1 to entry: More formally, a use case defines a set of use case instances or scenarios.

3.4463

use case diagram

1. UML diagram that shows actors, use cases, and their relationships

3.4464

use case model

1. model that describes a system's functional requirements in terms of use cases

3.4465

use case specification

1. document that describes a use case

Note 1 to entry: A use case specification's fundamental parts are the use case name, brief description, precondition, basic flow, postcondition, and alternate flow.

3.4466

use error

1. user action or lack of user action while using the interactive system that leads to a different result than that intended by the manufacturer or expected by the user [*ISO/IEC 25066:2016, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report*, 3.20]

Note 1 to entry: Use errors can result from a mismatch between the characteristics of the user, user interface, task, or use environment. Use error includes the inability of the user to complete a task. Users might be aware or unaware that a use error has occurred. A malfunction of an interactive system that causes an unexpected result is not considered a use error. An unexpected physiological response of the patient is not by itself considered a use error.

3.4467

use of IT

1. planning, design, development, deployment, operation, management, and application of IT to meet the needs of the organization

Note 1 to entry: includes both the demand for and the supply of IT services by internal groups, specialist IT units, or external suppliers and utility services (such as those providing software as services)

3.4468

usefulness

1. degree to which a user is satisfied with perceived achievement of pragmatic goals, including the results of use and the consequences of use [*ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.1.3.1]

3.4469

user

1. individual or group that interacts with a system or benefits from a system during its utilization [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.3.16; ISO/IEC TS 24748-1:2016 *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.60; ISO/IEC/IEEE 15288:2015 *Systems and software engineering — System life cycle processes*, 4.1.52] 2. person who interacts with a system, product or service [ISO/IEC 25064:2013 *Systems and software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: User needs report*, 4.17] 3. any person or thing that communicates or interacts with the software at any time [ISO/IEC 14143-1:2007 *Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts*; ISO/IEC 19761:2011 *Software engineering — COSMIC: a functional size measurement method*, 2.27; ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.50; ISO/IEC 29881:2010 *Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*, 3.9] 4. individual or group that benefits from a system during its utilization [ISO/IEC/IEEE 15939:2017 *Systems and software engineering — Measurement process*, 3.40] 5. person who performs one or more tasks with software; a member of a specific audience [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.54] 6. person (or instance) who uses the functions of a CBSS via a terminal (or an equivalent machine-user-interface) by submitting tasks and receiving the computed results [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.31] 7. person who derives engineering value through interaction with a CASE tool [IEEE 1175.2-2006 *IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections*, 3.17] 8. individual or organization that uses the system or software to perform a specific function [ISO/IEC 25000:2014 *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.40] 9. individual or group that benefits from a ready to use software product during its utilization [ISO/IEC 25051:2014 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.26]

cf. developer, end user, functional user, indirect user, operator, secondary user

EXAMPLE operators, recipients of the results of operating the system or software; a bank customer who visits a branch, receives a paper statement, or carries out telephone banking using a call center

Note 1 to entry: The user can perform other roles such as acquirer or maintainer. The role of user and the role of operator can be vested, simultaneously or sequentially, in the same individual or organization,

3.4470

user documentation

1. documentation for users of a system, including a system description and procedures for using the system to obtain desired results 2. information to describe, explain, or instruct how to use software [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.55] 3. information that is supplied with the software to help the users in their use of that software [ISO/IEC 25051:2014 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Ready to Use Software Product (RUSP) and instructions for testing*, 4.1.26]

cf. user manual

3.4471

user error protection

1. degree to which a system protects users against making errors [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.4.4]

3.4472

user experience

1. person's perceptions and responses that result from the use or anticipated use of a product, system or service [ISO/IEC TR 25060:2010 *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information*, 2.20]

Note 1 to entry: User experience is a consequence of brand image, presentation, functionality, system performance, interactive behavior, and assistive capabilities of the interactive system; the user's internal and physical state resulting from prior experiences, attitudes, skills and personality; and the context of use.

3.4473

user group

1. subset of intended users who are differentiated from other intended users by factors such as age, culture or expertise that are likely to influence usability [ISO/IEC 25062:2006 *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports*, 4.7]

3.4474

user interaction

1. exchange of information between a user and an interactive system via the user interface to complete the intended task [ISO/IEC TR 25060:2010 *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information*, 2.22]

Note 1 to entry: User interaction specifications focus on user interactions without considering implementation details.

3.4475

user interface

1. components of an interactive system (software or hardware) that provide information and controls for the user to accomplish specific tasks with the interactive system [ISO/IEC TR 25060:2010 *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information*, 2.23] **2.** ensemble of software and hardware that allows a user to interact with a computer system [ISO/IEC 26514:2008 *Systems and software engineering — requirements for designers and developers of user documentation*, 4.56] **3.** interface that enables information to be passed between a human user and hardware or software components of a computer system [ISO/IEC 19506:2012 *Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*, 4] **4.** all components of an interactive system (software or hardware) that provide information and controls for the user to accomplish specific tasks with the interactive system [ISO/IEC 25063:2014 *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description*]

3.4476

user interface aesthetics

1. degree to which a user interface enables pleasing and satisfying interaction for the user [ISO/IEC 25010:2011 *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*, 4.2.4.5]

Note 1 to entry: refers to properties of the product or system that increase the pleasure and satisfaction of the user, such as the use of color and the nature of the graphical design

3.4477

user interface element

user interface object

1. entity of the user interface that is presented to the user by the software [ISO/IEC TR 25060:2010 *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information*, 2.24]

EXAMPLE: a visual representation or an interaction mechanism for a task object (such as a letter, a sales order, electronic parts, or a wiring diagram) or a system object (such as a printer, hard disk, or network connection); basic objects (such as window title bars, menu items, push buttons, image maps, and editable text fields) or containers (such as windows, grouping boxes, menu bars, menus, groups of mutually exclusive option buttons, and compound images that are made up of several smaller images); messages and action prompts

Note 1 to entry: User interface elements can be interactive or not, and either entities relevant to the task or entities of the user interface

3.4478

user interface task

1. task that hides the details of the interface to and interacts sequentially with a human user

3.4479

user manual

user guide

1. document that presents the information necessary to employ a system or component to obtain desired results
2. document that describes how to use a functional unit, and that can include description of the rights and responsibilities of the user, the owner, and the supplier of the unit [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

cf. data input sheet, diagnostic manual, installation manual, operator manual, programmer manual, maintenance manual, support manual

Note 1 to entry: Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.) and those who use the system for its intended purpose.

3.4480

user need

1. prerequisite identified as necessary for a user, or a set of users, to achieve an intended outcome, implied or stated within a specific context of use [*ISO/IEC 25064:2013 Systems and software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: User needs report, 4.19*] 2. set of functional user requirements and non-functional user requirements that the users need the system to fulfill [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, A.13*]

3.4481

user profile

1. set of attributes that are unique to a specific user or user group, such as job function or subscription to a service, used to control the parts of the system or web page that users can access [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 4.26*]

3.4482

user requirements (UR)

usage requirements

1. requirements for use that provide the basis for design and evaluation of interactive systems to meet identified user needs [*ISO/IEC TR 25060:2010 Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: General framework for usability-related information, 2.21*] 2. description of the set of user needs for the software [*ISO/IEC 14143-1:2007 Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts, 3.12*] 3. expression of perceived need from individual or group that benefits from a system during its utilization [*ISO/IEC TR 24766:2009 Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities, 3.10*]

Note 1 to entry: User requirements specify the extent to which user needs and capabilities are to be met when using the system. They are not requirements on the users. User requirements are derived from user needs and capabilities in order to make use of the system in an effective, efficient, safe and satisfying manner. User requirements comprise two subsets: functional user requirements and non-functional user requirements. [ISO 25063:2014] In software-engineering terms, user requirements comprise both "functional" and "non-functional" requirements based on user needs and capabilities.

3.4483

user story

1. simple narrative illustrating the user goals that a software function will satisfy [*ISO/IEC/IEEE 26515: 2011 Systems and software engineering: Developing user documentation in an agile environment, 4.16*] 2. a narrative description of a software requirement, function, feature, or quality attribute, presented as a narrative of desired user interactions with a software system [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4484

user terminal

1. terminal that enables a user to communicate with a computer [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4485

user type

- 1.** classification of emulated users that is defined by the combination of 1) the relative frequencies of the use of chain types; 2) the preparation times (mean values and their standard deviations) [ISO/IEC 14756:1999 *Information technology — Measurement and rating of performance of computer-based software systems*, 4.32]

3.4486

user view

- 1.** Functional User Requirements as perceived by the user [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.52] **2.** the application as seen through the eyes of the user [ISO/IEC 24570:2005 *Software engineering — NESMA functional size measurement method version 2.1 — Definitions and counting guidelines for the application of Function Point Analysis*]

Note 1 to entry: Developers translate the user view into software to provide a solution.

3.4487

user-based evaluation

- 1.** evaluation that involves representative users performing tasks with the system to enable identification of usability problems or measurements of efficiency, effectiveness, user satisfaction, or other user experiences [ISO/IEC 25066:2016, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report*, 3.22]

3.4488

user-friendly

- 1.** pertaining to ease and convenience of use by humans [ISO/IEC 2382:2015, *Information technology — Vocabulary*] **2.** pertaining to a computer system, device, program, or document designed with ease of use as a primary objective

3.4489

user-recognizable

- 1.** of requirements for processes or data, agreed upon and understood by both the user and the software developer [ISO/IEC 20926:2009 *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*, 3.51]

3.4490

UTC

- 1.** Coordinated Universal Time [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 5]

3.4491

utility

- 1.** software tool designed to perform some frequently used support function **2.** measure of value within a given value system, often measured on a scale of 0 to 100

EXAMPLE: a program to copy magnetic tapes

3.4492

utilization

- 1.** in computer performance evaluation, a ratio representing the amount of time a system or component is busy divided by the time it is available
cf. busy time, idle time, up time

3.4493

utilization bound theorem

- 1.** real-time scheduling theorem stating the conditions under which a set of n independent periodic tasks scheduled by the rate-monotonic algorithm will always meet their deadlines

3.4494

V&V

- 1.** verification and validation.

3.4495

VAC

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

3.4496

valid

1. status of an information structure that follows the specified XML Schema document and is valid from an XML perspective [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.52]

3.4497

validate scope

1. the process of formalizing acceptance of the completed project deliverables [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4498

validated deliverables

1. deliverables that are result of executing quality control process to determine correctness [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4499

validated metric

1. metric whose values have been statistically associated with corresponding quality factor values [*IEEE 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*, 2.25]

3.4500

validation

1. confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.41; *ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*, 2.61; *ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes*, 4.1.53] **2.** process of providing evidence that the system, software, or hardware and its associated products satisfy requirements allocated to it at the end of each life cycle activity, solve the right problem (e.g., correctly model physical laws, implement business rules, and use the proper system assumptions), and, and satisfy intended use and user needs [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1.35] **3.** In a life cycle context, the set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals and objectives [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes*, 4.54] **4.** process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.1] **5.** the assurance that a product, service, or system meets the needs of the customer and other identified stakeholders. It often involves acceptance and suitability with external customers [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*] *cf. verification*

Note 1 to entry: Validation in a system life cycle context is the set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals, and objectives. The right system has been built. Validation demonstrates that the system can be used by the users for their specific tasks. "Validated" is used to designate the corresponding status. Multiple validations can be carried out if there are different intended uses.

3.4501

validation test

1. test to determine whether an implemented system fulfils its specified requirements [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4502

value

1. number or category assigned to an attribute of an entity by making a measurement [*ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*, 4.42] **2.** numerical or categorical result assigned to a base measure, derived measure, or indicator [*ISO/IEC/IEEE 15939:2017 Systems and software engineering — Measurement process*, 3.41] **3.** entity that

is possibly an actual parameter in a request [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.2.26]

3.4503

value baseline

1. measure of a set of assets before an optimization, assigning relevant values to each group of assets being tracked [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.53]

3.4504

value class

1. class that represents instances that are pure values [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.202]

Note 1 to entry: The constituent instances of a value class do not come and go and cannot change state.

3.4505

value engineering (VE)

1. an approach used to optimize project life cycle costs, save time, increase profits, improve quality, expand market share, solve problems, and/or use resources more effectively [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4506

value list constraint

1. constraint that specifies the set of all acceptable instance values for a value class [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.203]

3.4507

value range constraint

1. constraint that specifies the set of all acceptable instance values for a value class where the instance values are constrained by a lower and/or upper boundary [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.204]

EXAMPLE: Azimuth, which is required to be between -180° to +180°

Note 1 to entry: A range constraint only makes sense if there is a linear ordering specified.

3.4508

value-added reseller (VAR)

1. company licensed to repackage and support existing products, such as combined software packages [*ISO/IEC 19770-2:2015 Information technology — Software asset management — Part 2: Software identification tag*, 4.1.45]

3.4509

VAR

1. value-added reseller

3.4510

variability

1. characteristics that can differ among members of the product line [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.23]

3.4511

variability binding

1. act of determining the variant of the variation point defined in the variability model [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.7]

3.4512

variability constraint

1. constraint relationships between a variant and a variation point, between two variants, and between two variation points [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.24]

3.4513

variability dependency

1. relationship between a variation point and a set of variants, which indicates that the variation point implies a decision about the variants [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.25]

3.4514

variability documentation

1. detailed description of variability models being used across the member products within a product line [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.8]

3.4515

variability in requirements

1. external and internal variability in requirements engineering [*ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering*, 3.20]

3.4516

variability in space

1. variation that occurs at the same time with a different shape [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.9]

3.4517

variability in time

1. variation that occurs at different times [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.10]

3.4518

variability management

1. managerial tasks relate to variability and has two dimensions: variability dimension and asset dimension [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.26]

3.4519

variability mechanism

1. variability implementation methods in a product line for supporting assembly of domain assets [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.11]

3.4520

variability model

1. explicit definition for product line variability [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management*, 3.27]

3.4521

variability traceability

1. trace links established for a variability model with both domain assets and application assets where variants are bound [*ISO/IEC 26555:2015 Software and systems engineering — Tools and methods for product line technical management*, 3.12]

3.4522

variable

1. quantity or data item whose value can change **2.** instance whose identity is unknown at the time of writing [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.205] **3.** data item whose value can change during program execution

Note 1 to entry: A variable is represented by an identifier that begins with an upper-case letter.

3.4523

variable cost

1. cost, such as the cost of material, which is directly dependent on the rate of production
cf. fixed cost

3.4524

variable definition

- 1.** see data definition [*ISO/IEC/IEEE 29119-4:2015 Software and systems engineering — Software testing — Part 4: Test techniques, 4.28*]

3.4525

variable trace

data-flow trace

data trace

value trace

- 1.** record of the name and values of variables accessed or changed during the execution of a computer program
cf. execution trace, retrospective trace, subroutine trace, symbolic trace

3.4526

variance

- 1.** a quantifiable deviation, departure, or divergence away from a known baseline or expected value [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. variation

3.4527

variance analysis

- 1.** a technique for determining the cause and degree of difference between the baseline and actual performance [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4528

variance at completion (VAC)

- 1.** a projection of the amount of budget deficit or surplus, expressed as the difference between the budget at completion and the estimate at completion [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4529

variant

- 1.** in fault tolerance, a version of a program resulting from the application of software diversity **2.** one alternative that is used to realize particular variation points [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.27*]

3.4530

variation

- 1.** an actual condition that is different from the expected condition that is contained in the baseline plan [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]
cf. variance

3.4531

variation point

- 1.** representation corresponding to particular variable characteristics of products, domain assets, and application assets in the context of a product line [*ISO/IEC 26550:2015 Software and systems engineering — Reference model for product line engineering and management, 3.29*]

Note 1 to entry: In principle, each variation point has at least one variant.

3.4532

VCRM

- 1.** verification cross-reference matrix [*IEEE 15288.2:2014 IEEE Standard for Technical Reviews and Audits on Defense Programs, 3.2*]
cf. RTM

3.4533

VDD

- 1.** version description document

3.4534

velocity

1. a measure of a team's productivity rate at which the deliverables are produced, validated, and accepted within a pre-defined interval. Velocity is a capacity planning approach frequently used to forecast future project work. [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition] **2.** the rate of current work unit completion, measured as work units completed per fixed time period, such as story points, delivered features, functions, function points, user stories, use cases, or requirements completed in a given time period. Used as a measure of burndown rate or burnup rate [Software Extension to the PMBOK® Guide Fifth Edition]

3.4535

vendor branch

1. branch for keeping track of versions of imported software

Note 1 to entry: Differences between successive versions can then be readily applied to the locally modified import.

3.4536

verb phrase

1. part of the label of a relationship that names the relationship in a way that a sentence can be formed by combining the first class name, the verb phrase, the cardinality expression, and the second class name or role name [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.206] **2.** phrase used to name a relationship, which consists of a verb and words that constitute the object of the phrase [IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.206]

EXAMPLE: The statement "each project funds one or more tasks" could be derived from a relationship showing "project" as the first class, "task" as the second class with a "one or more" cardinality, and "funds" as the verb phrase.

Note 1 to entry: A verb phrase is ideally stated in active voice.

3.4537

verifiable

1. can be checked for correctness by a person or tool [ISO/IEC/IEEE 15289:2015 Systems and software engineering — Content of life-cycle information products (documentation), 5.27]

3.4538

verification

1. confirmation, through the provision of objective evidence, that specified requirements have been fulfilled [ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.55; ISO/IEC 25000:2014 Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE, 4.43; ISO/IEC TS 24748-1:2016 Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management, 2.62; ISO/IEC/IEEE 15288:2015 Systems and software engineering — System life cycle processes, 4.1.54] **2.** the evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition] **3.** process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase [IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.36] **4.** process of providing objective evidence that the system, software, or hardware and its associated products conform to requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance), satisfy standards, practices, and conventions during life cycle processes, and successfully complete each life cycle activity and satisfy all the criteria for initiating succeeding life cycle activities [IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1] *cf.* validation

Note 1 to entry: Verification in a life cycle context is a set of activities that compares a product of the life cycle against the required characteristics for that product. This can include, but is not limited to, specified requirements, design description, and the system itself. The system has been built right. "Verified" is used to designate the corresponding status. Verification of interim work products is essential for proper understanding and assessment of the life cycle phase product(s). A system could be verified to meet the stated requirements, yet be unsuitable for operation by the actual users.

3.4539

verification and validation (V&V)

1. process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements
cf. independent verification and validation

3.4540

verification and validation (V&V) effort

1. work associated with performing the V&V processes, activities, and tasks [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation, 3.1.37*]

3.4541

verification method

1. a method that tests an FSM method, and provides objective evidence of the extent to which a particular performance property is exhibited [*ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods, 3.1*]

Note 1 to entry: Verification of an FSM method should produce a result that indicates the extent to which a performance property is exhibited, or whether a performance property is exhibited to a stated extent. For this reason, there is no concept of "pass" or "fail". An FSM method can be considered to be either "verified" or "not verified", for a particular performance property, based on whether or not the appropriate verification has been conducted.

3.4542

verification sponsor

1. the person or organization that requires the verification to be performed and provides financial or other resources to carry it out [*ISO/IEC TR 14143-3:2003 Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods, 3.11*]

3.4543

verification test

1. test of a system to prove that it meets all its specified requirements at a particular stage of its development. [*ISO/IEC 2382:2015, Information technology — Vocabulary 20.05.03*]

3.4544

Verilog hardware description language (VHDL)

1. hardware description language used to design and verify digital circuits

Note 1 to entry: standardized in IEEE 1394

3.4545

version

1. initial release or re-release of a computer software configuration item, associated with a complete compilation or recompilation of the computer software configuration item [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*] 2. initial release or complete re-release of a document, as opposed to a revision resulting from issuing change pages to a previous release [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering, 2.1*] 3. operational software product that differs from similar products in terms of capability, environmental requirements, and configuration 4. identified instance of a configuration item [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities, 3.15*] 5. identified instance of an item [*ISO/IEC 12207:2008 Systems and software engineering — Software life cycle processes, 4.56*] 6. unique string of number and letter values indicating a unique revision of an item [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary, 3.54*]
cf. release

Note 1 to entry: Versions often identify revisions of software that provide unique functionality or fixes. A version typically has multiple parts, such as a major version, indicating large changes in functionality or user interface changes, and a minor version, indicating smaller changes in functionality or user interface changes.

3.4546

version control

- 1.** establishment and maintenance of baselines and the identification and control of changes to baselines that make it possible to return to the previous baseline
cf. change control

3.4547

version description document (VDD)

- 1.** document that accompanies and identifies a given version of a system or component

Note 1 to entry: Typical contents include an inventory of system or component parts, identification of changes incorporated into this version, and installation and operating information unique to the version described.

3.4548

version identifier

- 1.** supplementary information used to distinguish a version of a configuration item from other versions [*ISO/IEC TR 18018:2010 Information technology — Systems and software engineering — Guide for configuration management tool capabilities*, 3.16]

Note 1 to entry: Version numbers are used to identify the version of the software product being compared with another version.

3.4549

versioning

- 1.** assignment of either unique version names or unique version numbers to unique states of software configuration items, usually for a specific purpose, such as a release of the software product to an external group or the identification of a specific baseline [*IEEE 828-2012 IEEE Standard for Configuration Management in Systems and Software Engineering*, 2.1]

3.4550

vertical microinstruction

- 1.** microinstruction that specifies one of a sequence of operations needed to carry out a machine language instruction
cf. diagonal microinstruction, horizontal microinstruction

Note 1 to entry: Vertical microinstructions are relatively short, 12 to 24 bits, and are called 'vertical' because a sequence of such instruction, normally listed vertically on a page, is required to carry out a single machine language instruction.

3.4551

very small entity (VSE)

- 1.** entity (enterprise, organization, department or project) having up to 25 people [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 2.41]

3.4552

VHSIC

- 1.** very high speed integrated circuit

3.4553

video display terminal (VDT)

visual display terminal

visual display unit (VDU)

- 1.** user terminal with a display screen and usually equipped with an input unit such as a keyboard [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4554

video random access memory (VRAM)

- 1.** random access memory designed for the frame buffers of graphics cards

3.4555

view

- 1.** developer's copy of a branch **2.** collection of subject domains, classes, relationships, responsibilities, properties, constraints, and notes assembled or created for a certain purpose and covering a certain scope [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.207*] **3.** collection of entities and assigned attributes (domains) assembled for some purpose [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.207*] **4.** set of related categories **5.** representation of a whole system from the perspective of a related set of concerns [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

Note 1 to entry: A view can cover the entire area being modeled or only a part of that area.

3.4556

view diagram

- 1.** graphic representation of the underlying semantics of a view [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject), 3.1.208*]

3.4557

viewpoint

- 1.** specification of the conventions for constructing and using a view [*ISO/IEC 19506:2012 Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM), 4*]

Note 1 to entry: A pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

3.4558

viewpoint (on a system)

- 1.** form of abstraction achieved using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within a system [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 3.2.7*]

3.4559

viewpoint correspondence

- 1.** statement that some terms or other linguistic constructs in a specification from one viewpoint are associated with (e.g., describe the same entities as) terms or constructs in a specification from a second viewpoint [*ISO/IEC 10746-2:2009 Information technology — Open Distributed Processing — Reference Model: Foundations, 3.2.8*]

Note 1 to entry: The forms of association that can be expressed will depend on the specification technique used. The terms associated by a correspondence need not necessarily be expressed using a single specification technique. The correspondence can associate a term in one specification technique with a term in some different specification technique. Rather than linking every individual pair of terms, general correspondences can also be expressed between specification techniques themselves. For example, composition operators defined in different specification techniques can be associated, implying correspondences wherever these operators are used to link terms in the respective viewpoints.

3.4560

viewpoint statement

- 1.** brief statement of the perspective of an IDEF0 model that is presented in the a-0 context diagram of the model [*IEEE 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0, 2.1.109*]

3.4561

violation

- 1.** behavior contrary to that required by a rule [*ISO/IEC 15414:2015 Information technology — Open distributed processing — Reference model — Enterprise language, 6.3.8*] **2.** behavior, act, or event deviating from a system's desired property or claim of interest [*ISO/IEC 15026-1:2013 Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary, 3.4.7*]

Note 1 to entry: In the area of safety, the term "violation" is used to refer to a deliberate human contravention of a procedure or rule. A rule or policy can provide behavior to occur upon violation of that or some other rule or policy.

3.4562

virtual

1. pertaining to a functional unit that appears to be real, but whose functions are accomplished by other means [*ISO/IEC 2382:2015, Information technology — Vocabulary*] **2.** for an entity, being composed of one or more underlying base entities [*ISO/IEC 15476-4:2005 Information technology — CDIF semantic metamodel — Part 4: Data models*, 6.16]

3.4563

virtual address

1. in a virtual storage system, the address assigned to an auxiliary storage location to allow that location to be accessed as though it were part of main storage

cf. real address

3.4564

virtual machine (VM)

1. virtual data processing system that appears to be at the disposal of a particular user, but whose functions are accomplished by sharing the resources of a real data processing system [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

3.4565

virtual reference

1. references made to concepts other than specific meta-entities in a metamodel [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview*, 4.2]

Note 1 to entry: represented by boxes with diagonal striping.

3.4566

virtual storage

multilevel storage

virtual memory

1. storage allocation technique in which auxiliary storage can be addressed as though it were part of main storage
cf. real storage, virtual address, paging (2)

Note 1 to entry: Portions of a user's program and data are placed in auxiliary storage, and the operating system automatically swaps them in and out of main storage as needed.

3.4567

virtual team

1. a team that is separated by geography or work schedules and maintains electronic communication [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4568

visibility

1. degree to which a transaction can access object state concurrently with other transactions [*ISO/IEC 10746-3:2009 Information technology — Open Distributed Processing — Reference Model: Architecture*, 13.7.1.3] **2.** specification, for a property, of "who can see it?" [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.209]

Note 1 to entry: Visibility is private, protected, or public.

3.4569

vocabulary

1. collection of information related to a specific subset of terms related to a specific domain [*ISO/IEC 25024:2015 Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Measurement of data*, 4.38]

Note 1 to entry: Vocabulary is generally used to keep consistency, to avoid duplication, and to support synonyms.

3.4570

voice of the customer

1. a planning technique used to provide products, services, and results that truly reflect customer requirements by translating those customer requirements into the appropriate technical requirements for each phase of project product development [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4571

volatile memory

1. unit that stores data only while power is on
cf. non-volatile memory

3.4572

volume testing

1. type of performance efficiency testing conducted to evaluate the capability of the test item to process specified volumes of data (usually at or near maximum specified capacity) in terms of throughput capacity, storage capacity, or both [*ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions*, 4.95]

3.4573

VRAM

1. video random access memory

3.4574

VSCID

1. vendor service context codeset ID [*ISO/IEC 19500-2:2012 Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*, 3.3]

3.4575

VSE

1. very small entity [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*, 3.2; *ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile*, 4.2]

Note 1 to entry: The plural is VSEs (very small entities)

3.4576

VVP

1. verification and validation plan [*IEEE 1012-2012 IEEE Standard for System and Software Verification and Validation*, 3.2]

3.4577

W3C

1. World Wide Web Consortium [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.2]

3.4578

waiver

1. written authorization to accept a configuration item or other designated item which, during production or after having been submitted for inspection, is found to depart from specified requirements, but is nevertheless considered suitable for use as is or after rework by an approved method
cf. configuration control, deviation, engineering change

3.4579

walk-through

1. static analysis technique in which a designer or programmer leads members of the development team and other interested parties through a segment of documentation or code, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems

3.4580

WAP

1. Wireless Application Protocol [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.4581

war room

1. room used for project conferences and planning, often displaying charts of cost, schedule status, and other key project data.

3.4582

warning

1. advisory information in documentation that states that performing some action can lead to serious or dangerous consequences [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation, 4.57*]

cf. caution, note

3.4583

watchdog timer (WDT)

1. electronic unit that triggers a reset of an embedded system or other corrective action if the main program, due to some fault condition, fails to periodically signal it

3.4584

waterfall model

1. model of the software development process in which the constituent activities, typically a concept phase, requirements phase, design phase, implementation phase, test phase, and installation and checkout phase, are performed in that order, possibly with overlap but with little or no iteration

cf. incremental development, rapid prototyping, spiral model

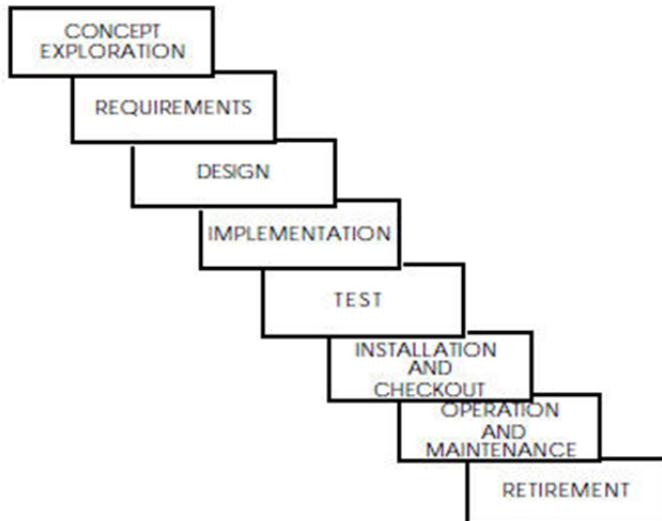


Figure 19 — Waterfall model

3.4585

WBS

1. work breakdown structure [*ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 4.2; ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 5*]

3.4586

WCAG

1. Web Content Accessibility Guidelines [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information, 5*]

3.4587

WDT

1. watchdog timer

3.4588

wearout-failure period

1. period in the life cycle of a system or component during which hardware failures occur at an increasing rate due to deterioration

cf. constant-failure period, early-failure period, bathtub curve

3.4589

web page

1. coherent presentation of a set of content objects and associated interaction objects delivered to users through a browser in accordance with Internet protocols [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.28]

Note 1 to entry: A web page can be generated dynamically from the server side, and can incorporate multimedia, applets, or other elements active on either the client or server side.

3.4590

webmaster

1. person or group responsible to the website owner for ongoing maintenance of the site's presentation and availability [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.27]

3.4591

website

1. collection of logically connected web pages managed as a single entity [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.29]

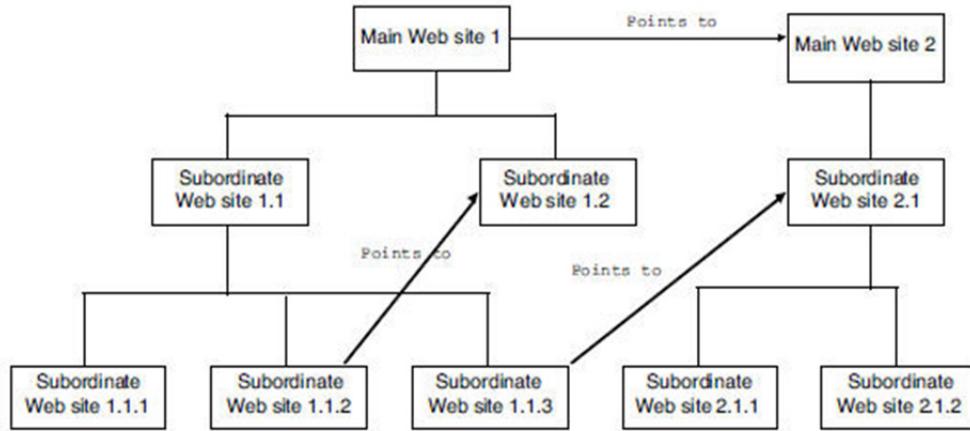


Figure 20 — Website

3.4592

website owner

1. organization responsible for the site content and site design [ISO/IEC/IEEE 23026:2015 *Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.30]

Note 1 to entry: The website owner can select a supplier as the website provider or can also be the website provider.

3.4593

website provider

1. organization responsible for operation of the website and delivery of site content to users [*ISO/IEC/IEEE 23026:2015 Systems and software engineering — Engineering and management of websites for systems, software, and services information*, 4.31]

Note 1 to entry: The website provider can also be the site owner, webmaster, site designer, or the internet service provider for the site.

3.4594

weighted milestone method

1. an earned value method that divides a work package into measurable segments, each ending with an observable milestone, and then assigns a weighted value to the achievement of each milestone [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4595

what-if scenario analysis

1. the process of evaluating scenarios in order to predict their effect on project objectives [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4596

WHILE

pretested iteration

1. single-entry, single-exit loop in which the loop control is executed before the loop body
cf. closed loop, UNTIL, leading decision

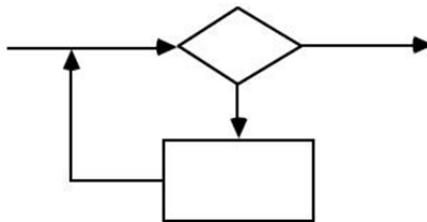


Figure 21 — WHILE construct

3.4597

whitespace

1. nondisplaying formatting characters that are embedded within a block of free text [*IEEE 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*, 3.1.210]

EXAMPLE: spaces and tabs

3.4598

window

1. area with visible boundaries that presents a view of a software object or through which a user conducts a dialog with a computer system [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.58]

3.4599

wizard

1. procedural form of help that guides a user through each step of a task through dialog with the user [*ISO/IEC 26514:2008 Systems and software engineering — requirements for designers and developers of user documentation*, 4.59]

3.4600

word

1. sequence of bits or characters that is stored, addressed, transmitted, and operated on as a unit within a given computer **2.** element of computer storage that can hold a sequence of bits or characters as in (1) **3.** sequence of bits or characters that has meaning and is considered an entity in some language
cf. computer word

3.4601

work authorization

1. a permission and direction, typically written, to begin work on a specific schedule activity or work package or control account. It is a method for sanctioning project work to ensure that the work is done by the identified organization, at the right time, and in the proper sequence. [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4602

work authorization system

1. a subsystem of the overall project management system. It is a collection of formal documented procedures that defines how project work will be authorized (committed) to ensure that the work is done by the identified organization, at the right time, and in the proper sequence. It includes the steps, documents, tracking system, and defined approval levels needed to issue work authorizations. [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4603

work breakdown structure (WBS)

1. a hierarchical decomposition of the total scope of work to be carried out by the project team to accomplish the project objectives and create the required deliverables [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition] **2.** deliverable-oriented hierarchical decomposition of the work to be executed by the project team to accomplish the project objectives and create the required deliverables [ISO/IEC TR 29110-5-6-2:2014 Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-6-2: Systems engineering — Management and engineering guide: Generic profile group: Basic profile, 3.12]

3.4604

work breakdown structure component

1. an entry in the work breakdown structure that can be at any level [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4605

work breakdown structure dictionary

1. a document that provides detailed deliverable, activity, and scheduling information about each component in the work breakdown structure [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4606

work effort

1. labor resources required for the production of a specified output

Note 1 to entry: Labor resources are usually expressed as work hours.

3.4607

work package

1. the work defined at the lowest level of the work breakdown structure for which cost and duration can be estimated and managed [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4608

work performance data

1. the raw observations and measurements identified during activities being performed to carry out the project work [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4609

work performance information

1. the performance data collected from various controlling processes, analyzed in context and integrated based on relationships across areas [A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition]

3.4610

work performance reports

performance reporting
performance reports

1. the physical or electronic representation of work performance information compiled in project documents, intended to generate decisions, actions, or awareness [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4611

work product

1. artifact resulting from the execution of a process [*IEEE 730-2014 IEEE Standard for Software Quality Assurance Processes, 3.2*] **2.** any artifact produced by a process [*ISO/IEC 15940:2013 Systems and software engineering — Software Engineering Environment Services, 2.5*] **3.** artifact associated with the execution of a process [*ISO/IEC TR 29110-1:2016 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview, 3.77*]

EXAMPLE: the project plan, supporting process requirements, design documentation, source code, test plans, meeting minutes, schedules, budgets, and problem reports

Note 1 to entry: Some subset of the work products will be baselined and some will form the set of project deliverables.

3.4612

work unit

1. a project task such as constructing or testing a function point, user story, feature, or requirement [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4613

workaround

1. a response to a threat that has occurred, for which a prior response had not been planned or was not effective [*A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*]

3.4614

workflow board

kanban board

1. in software development, a visual representation of work for developers who pull tasks from the task backlog; used for on-demand or resource-bound scheduling [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4615

working metamodel

1. definition of the specific meta-objects that can be instantiated in the model section of a CDIF transfer [*ISO/IEC 15474-1:2002 Information technology — CDIF framework — Part 1: Overview, 4.2*]

Note 1 to entry: The working metamodel comprises the meta-objects in the CDIF semantic metamodel that are used by the subject areas referenced in the metamodel section of the transfer, and the meta-objects defined as extensions in the metamodel section.

3.4616

working set

1. in the paging method of storage allocation, the set of pages that are most likely to be resident in main storage at any given point of a program's execution

3.4617

working space

working area

working storage

1. that portion of main storage that is assigned to a computer program for temporary storage of data

3.4618

workload

1. mix of tasks typically run on a given computer system

Note 1 to entry: Major characteristics include input/output requirements, amount and kinds of computation, and computer resources required.

3.4619

workload model

1. model used in computer performance evaluation, depicting resource utilization and performance measures for anticipated or actual workloads in a computer system

cf. system model

3.4620

workstation

1. functional unit that usually has special purpose computing capabilities and includes user-oriented input units and output units [*ISO/IEC 2382:2015, Information technology — Vocabulary*]

EXAMPLE: a programmable terminal, a nonprogrammable terminal or a standalone microcomputer

3.4621

write

write type

1. to record data in a storage device or on a data medium **2.** data movement that moves a data group lying inside the functional process to persistent storage [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 2.28*]

cf. read

3.4622

write (-type)

1. a data movement type that moves a data group lying inside the functional process to persistent storage [*ISO/IEC 19761:2011 Software engineering — COSMIC: a functional size measurement method, 3.26*]

Note 1 to entry: A Write is considered to include certain associated data manipulations necessary to achieve the Write.

3.4623

writer

author

1. person designing or developing user documentation [*ISO/IEC/IEEE 26515: 2011 Systems and software engineering: Developing user documentation in an agile environment, 4.17*]

3.4624

writing reference

1. data storage entity or other record, or interface record to another software or system to which data is written in a BFC [*ISO/IEC 29881:2010 Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method, 3.10*]

Note 1 to entry: The number of writing references is greater than 0 with all BFC types where it is applicable.

3.4625

XFN

1. X/Open Federated Naming [*ISO/IEC 14771:1999 Information technology — Open Distributed Processing — Naming framework, 4*]

3.4626

XHTML

1. Extended HyperText Markup Language [*ISO/IEC/IEEE 16326:2009 Systems and software engineering — Life cycle processes — Project management, 5*]

3.4627

XMI

1. XML Metadata Interchange [*ISO/IEC 19500-3:2012 Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components, 4.3; ISO/IEC 19793:2015 Information technology — Open Distributed Processing — Use of UML for ODP system specifications, 4*]

3.4628

XML

1. extensible markup language [*ISO/IEC 15909-2:2011 Software and system engineering — High-level Petri nets — Part 2: Transfer format*, 4.2.9]

3.4629

XML schema definition

1. XML-based language that specifies a set of rules and structure for the creation of XML documents **2.** language that describes the structure of XML information [*ISO/IEC 19770-3:2016, Information technology — IT asset management — Part 3: Entitlement schema*, 3.1.42]

Note 1 to entry: XML documents follow all rules defined in an XSD definition in order to be considered a valid document.

3.4630

XML schema document (XSD)

1. document that describes the structure of XML information [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.55]

3.4631

XSD

1. XML schema definition **2.** XML schema document [*ISO/IEC 19770-5:2015 Information technology — IT asset management — Part 5: Overview and vocabulary*, 3.55]

3.4632

yesterday's weather

1. a report of work performance in the most recent reporting period [*Software Extension to the PMBOK® Guide Fifth Edition*]

3.4633

zero-address instruction

1. computer instruction that contains no address fields

cf. one-address instruction, two-address instruction, three-address instruction, four-address instruction

Annex A (informative)

List of References

The systems and software engineering standards and publications within the scope of ISO JTC 1/SC7 and the IEEE Computer Society refer to definitions from other standards maintained by other ISO or IEC Technical Committees. These are listed here as references for ISO/IEC/IEEE 24765.

- [1] IEC 61508-4, Functional safety of electrical/electronic/programmable electronic safety-related systems — Part 4: Definitions and abbreviations
- [2] ISO 9000:2015, *Quality management systems — Fundamentals and vocabulary*
- [3] ISO 9241-11:1998, Ergonomic requirements for office work with visual display terminals (VDTs) — Part 11: Guidance on usability
- [4] ISO 9241-110:2006, *Ergonomics of human-system interaction — Part 110: Dialogue principles*
- [5] ISO 9241-171:2008, *Ergonomics of human-system interaction — Part 171: Guidance on software accessibility*
- [6] ISO Guide 73:2009, *Risk management — Vocabulary*
- [7] ISO/IEC Guide 2:2004, *Standardization and related activities — General vocabulary*
- [8] ISO/IEC Guide 51:2014, *Safety aspects — Guidelines for their inclusion in standards*
- [9] ISO/IEC Guide 99:2007, *International vocabulary of metrology — Basic and general concepts and associated terms*

Bibliography

- [1] *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fifth Edition*
- [2] IEEE 1012-2012, *IEEE Standard for System and Software Verification and Validation*
- [3] IEEE 1016-2009, *IEEE standard for Information Technology-Systems Design-Software Design Descriptions*
- [4] IEEE 1028-2008, *IEEE Standard for Software Reviews and Audits*
- [5] IEEE 1044-2009, *IEEE Standard Classification for Software Anomalies*
- [6] IEEE 1045-1992 (R2002), *IEEE Standard for Software Productivity Metrics*
- [7] IEEE 1061-1998 (R2004), *IEEE Standard for Software Quality Metrics Methodology*
- [8] IEEE 1062-2015, *IEEE Recommended Practice for Software Acquisition*
- [9] IEEE 1074-2006, *IEEE Standard for Developing a Software Project Life Cycle Process*
- [10] IEEE 1175.1-2002 (R2007), *IEEE Guide for CASE Tool Interconnections-Classification and Description*
- [11] IEEE 1175.2-2006, *IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections*
- [12] IEEE 1175.3-2004, *IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying Software Behavior*
- [13] IEEE 1175.4-2008, *IEEE Standard for CASE Tool Interconnections - Reference Model for Specifying System Behavior*
- [14] IEEE 1228-1994 (R2002), *IEEE Standard for Software Safety Plans*
- [15] IEEE 1320.1-1998, (R2004), *IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0*
- [16] IEEE 1320.2-1998 (R2004), *IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*
- [17] IEEE 14764-2006, *Software Engineering - Software Life Cycle Processes - Maintenance*
- [18] IEEE 1517-2010, *IEEE Standard for Information Technology — System and software life cycle processes — Reuse processes*
- [19] IEEE 15288.1:2014, *IEEE Standard for Application of Systems Engineering on Defense Programs*
- [20] IEEE 15288.2:2014, *IEEE Standard for Technical Reviews and Audits on Defense Programs*
- [21] IEEE 730-2014, *IEEE Standard for Software Quality Assurance Processes*
- [22] IEEE 828-2012, *IEEE Standard for Configuration Management in Systems and Software Engineering*
- [23] IEEE 982.1-2005, *IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*
- [24] ISO 3535:1977, *Forms design sheet and layout chart*
- [25] ISO 5806:1984, *Information processing — Specification of single-hit decision tables*
- [26] ISO 5807:1985, *Information processing — Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts*
- [27] ISO/IEC 10746-1:1998, *Information technology — Open Distributed Processing — Reference model: Overview*

- [28] ISO/IEC 10746-2:2009, *Information technology — Open Distributed Processing — Reference Model: Foundations*
- [29] ISO/IEC 10746-3:2009, *Information technology — Open Distributed Processing — Reference Model: Architecture*
- [30] ISO/IEC 11411:1995, *Information technology — Representation for human communication of state transition of software*
- [31] ISO/IEC 12207:2008, *Systems and software engineering — Software life cycle processes*
- [32] ISO/IEC 13235-3:1998, *Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service*
- [33] ISO/IEC 14102:2008, *Information Technology — Guideline for the evaluation and selection of CASE tools*
- [34] ISO/IEC 14143-1:2007, *Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts*
- [35] ISO/IEC 14143-2:2011, *Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1*
- [36] ISO/IEC 14143-6:2012, *Information technology — Software measurement — Functional size measurement — Part 6: Guide for use of ISO/IEC 14143 series and related International Standards*
- [37] ISO/IEC 14568:1997, *Information technology — DXL: Diagram eXchange Language for tree-structured charts*
- [38] ISO/IEC 14598-5:1998, *Information technology — Software product evaluation — Part 5: Process for evaluators*
- [39] ISO/IEC 14752:2000, *Information technology — Open Distributed Processing — Protocol support for computational interactions*
- [40] ISO/IEC 14753:1999, *Information technology — Open Distributed Processing — Interface references and binding*
- [41] ISO/IEC 14756:1999, *Information technology — Measurement and rating of performance of computer-based software systems*
- [42] ISO/IEC 14769:2001, *Information technology — Open Distributed Processing — Type Repository Function*
- [43] ISO/IEC 14771:1999, *Information technology — Open Distributed Processing — Naming framework*
- [44] ISO/IEC 15026-1:2013, *Systems and software engineering — Systems and software assurance — Part 1: Concepts and vocabulary*
- [45] ISO/IEC 15026-3:2015, *Systems and software engineering — Systems and software assurance — Part 3: System integrity levels*
- [46] ISO/IEC 15414:2015, *Information technology — Open distributed processing — Reference model — Enterprise language*
- [47] ISO/IEC 15474-1:2002, *Information technology — CDIF framework — Part 1: Overview*
- [48] ISO/IEC 15474-2:2002, *Information technology — CDIF framework — Part 2: Modelling and extensibility*
- [49] ISO/IEC 15475-2:2002, *Information technology — CDIF transfer format — Part 2: Syntax SYNTAX.1*
- [50] ISO/IEC 15475-3:2002, *Information technology — CDIF transfer format — Part 3: Encoding ENCODING.1*
- [51] ISO/IEC 15476-4:2005, *Information technology — CDIF semantic metamodel — Part 4: Data models*
- [52] ISO/IEC 15909-1:2004, *Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation*
- [53] ISO/IEC 15909-2:2011, *Software and system engineering — High-level Petri nets — Part 2: Transfer format*

ISO/IEC/IEEE 24765:2017(E)

- [54] ISO/IEC 15940:2013, *Systems and software engineering — Software Engineering Environment Services*
- [55] ISO/IEC 16085:2006, *Systems and software engineering — Life cycle processes — Risk management*
- [56] ISO/IEC 16350:2015, *Information technology — Systems and software engineering — Application management*
- [57] ISO/IEC 19500-1:2012, *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 1: Interfaces*
- [58] ISO/IEC 19500-2:2012, *Information technology — Object Management Group — Common Object Request Broker Architecture (CORBA) — Part 2: Interoperability*
- [59] ISO/IEC 19500-3:2012, *Information technology — Object Management Group — Common Architecture Request Broker Architecture (CORBA) — Part 3: Components*
- [60] ISO/IEC 19506:2012, *Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)*
- [61] ISO/IEC 19761:2011, *Software engineering — COSMIC: a functional size measurement method*
- [62] ISO/IEC 19770-1:2012, *Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*
- [63] ISO/IEC 19770-2:2015, *Information technology — Software asset management — Part 2: Software identification tag*
- [64] ISO/IEC 19770-3:2016, *Information technology — IT asset management — Part 3: Entitlement schema*
- [65] ISO/IEC 19770-5:2015, *Information technology — IT asset management — Part 5: Overview and vocabulary*
- [66] ISO/IEC 19793:2015, *Information technology — Open Distributed Processing — Use of UML for ODP system specifications*
- [67] ISO/IEC 20926:2009, *Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009*
- [68] ISO/IEC 20968:2002, *Software engineering — Mk II Function Point Analysis — Counting Practices Manual*
- [69] ISO/IEC 2382-2015, *Information technology — Vocabulary*
- [70] 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*
- [71] ISO/IEC 24744:2014, *Software Engineering — Metamodel for development methodologies*
- [72] ISO/IEC 24773:2008, *Software engineering — Certification of software engineering professionals — Comparison framework*
- [73] ISO/IEC 25000:2014, *Systems and software Engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*
- [74] ISO/IEC 25001:2014, *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) — Planning and management*
- [75] ISO/IEC 25010:2011, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*
- [76] ISO/IEC 25020:2007, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Measurement reference model and guide*

- [77] ISO/IEC 25021:2012, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements*
- [78] ISO/IEC 25023:2016, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of system and software product quality*
- [79] ISO/IEC 25024:2015, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Measurement of data*
- [80] ISO/IEC 25040:2011, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation process*
- [81] ISO/IEC 25041:2012, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation guide for developers, acquirers and independent evaluators*
- [82] ISO/IEC 25045:2010, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Evaluation module for recoverability*
- [83] 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*
- [84] ISO/IEC 25062:2006, *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports*
- [85] ISO/IEC 25063:2014, *Systems and software engineering — Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description*
- [86] ISO/IEC 25064:2013, *Systems and software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability: User needs report*
- [87] ISO/IEC 25066:2016, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for Usability — Evaluation Report*
- [88] ISO/IEC 26513:2009, *Systems and software engineering — Requirements for testers and reviewers of user documentation*
- [89] ISO/IEC 26514:2008, *Systems and software engineering — requirements for designers and developers of user documentation*
- [90] ISO/IEC 26550:2015, *Software and systems engineering — Reference model for product line engineering and management*
- [91] ISO/IEC 26551:2016, *Software and systems engineering — Tools and methods for product line requirements engineering*
- [92] ISO/IEC 26555:2015, *Software and systems engineering — Tools and methods for product line technical management*
- [93] ISO/IEC 29110-2-1:2015, *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-1: Framework and taxonomy*
- [94] ISO/IEC 29155-1:2011, *Systems and software engineering — Information technology project performance benchmarking framework — Part 1: Concepts and definitions*
- [95] ISO/IEC 29155-2:2013, *Systems and software engineering — Information technology project performance benchmarking framework — Part 2: Requirements for benchmarking*
- [96] ISO/IEC 29155-3:2015, *Systems and software engineering — Information technology project performance benchmarking framework — Part 3: Guidance for reporting*
- [97] ISO/IEC 29881:2010, *Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*

- [98] ISO/IEC 30103:2015, *Software and Systems Engineering — Lifecycle Processes — Framework for Product Quality Achievement*
- [99] ISO/IEC 30130:2016, *Software engineering — Capabilities of software testing tools*
- [100] ISO/IEC 33001:2015, *Information technology — Process assessment — Concepts and terminology*
- [101] ISO/IEC 33003:2015, *Information technology — Process assessment — Requirements for process measurement frameworks*
- [102] ISO/IEC 33020:2015, *Information technology — Process assessment — Process measurement framework for assessment of process capability*
- [103] ISO/IEC 8631:1989, *Information technology — Program constructs and conventions for their representation*
- [104] ISO/IEC 90003:2014, *Software engineering — Guidelines for the application of ISO 9001:2008 to computer software*
- [105] ISO/IEC TR 12182:2015, *Systems and software engineering — Framework for categorization of IT systems and software, and guide for applying it*
- [106] ISO/IEC TR 14143-3:2003, *Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods*
- [107] ISO/IEC TR 14143-4:2002, *Information technology — Software measurement — Functional size measurement — Part 4: Reference model*
- [108] ISO/IEC TR 14143-5:2004, *Information technology — Software measurement — Functional size measurement — Part 5: Determination of functional domains for use with functional size measurement*
- [109] ISO/IEC TR 14471:2007, *Information technology — Software engineering — Guidelines for the adoption of CASE tools*
- [110] ISO/IEC TR 14759:1999, *Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use*
- [111] ISO/IEC TR 15846:1998, *Information technology — Software life cycle processes — Configuration Management*
- [112] ISO/IEC TR 18018:2010, *Information technology — Systems and software engineering — Guide for configuration management tool capabilities*
- [113] ISO/IEC TR 19759:2016, *Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)*
- [114] ISO/IEC TR 24766:2009, *Information technology — Systems and software engineering — Guide for requirements engineering tool capabilities*
- [115] 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*
- [116] ISO/IEC TR 29110-1:2016, *Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 1: Overview*
- [117] ISO/IEC TR 29110-2-2:2011, *Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 2-2: Guide for the development of domain-specific profiles*
- [118] ISO/IEC TR 29110-3-1:2015, *Systems and software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 3-1: Assessment Guide*
- [119] 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*

ISO/IEC/IEEE 24765:2017(E)

- [120] 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*
- [121] ISO/IEC TR 29154:2013, *Software engineering — Guide for the application of ISO/IEC 24773:2008 (Certification of software engineering professionals — Comparison framework)*
- [122] ISO/IEC TR 33014:2013, *Information technology — Process assessment — Guide for process improvement*
- [123] ISO/IEC TR 90005:2008, *Systems engineering — Guidelines for the application of ISO 9001 to system life cycle processes*
- [124] ISO/IEC TS 15504-10:2011, *Information technology — Process assessment — Part 10: Safety extension*
- [125] ISO/IEC TS 24748-1:2016, *Systems and software engineering — Life cycle management — Part 1: Guide for life cycle management*
- [126] ISO/IEC/IEEE 15288:2015, *Systems and software engineering — System life cycle processes*
- [127] ISO/IEC/IEEE 15289:2015, *Systems and software engineering — Content of life-cycle information products (documentation)*
- [128] ISO/IEC/IEEE 15939:2017, *Systems and software engineering — Measurement process*
- [129] ISO/IEC/IEEE 16326:2009, *Systems and software engineering — Life cycle processes — Project management*
- [130] ISO/IEC/IEEE 23026:2015, *Systems and software engineering — Engineering and management of websites for systems, software, and services information*
- [131] ISO/IEC/IEEE 24748-4:2016, *Systems and software engineering — Life cycle management-Part 4: Systems engineering planning*
- [132] ISO/IEC/IEEE 26511:2011, *Systems and software engineering — Requirements for managers of user documentation*
- [133] ISO/IEC/IEEE 26512:2011, *Systems and software engineering — Requirements for acquirers and suppliers of user documentation*
- [134] ISO/IEC/IEEE 26515:2011, *Systems and software engineering: Developing user documentation in an agile environment*
- [135] ISO/IEC/IEEE 29119-1:2013, *Software and systems engineering — Software testing — Part 1: Concepts and definitions*
- [136] ISO/IEC/IEEE 29119-2:2013, *Software and systems engineering — Software testing — Part 2: Test processes*
- [137] ISO/IEC/IEEE 29119-3:2013, *Software and systems engineering — Software testing — Part 3: Test documentation*
- [138] ISO/IEC/IEEE 29119-4:2015, *Software and systems engineering — Software testing — Part 4: Test techniques*
- [139] ISO/IEC/IEEE 29148:2011, *Systems and software engineering — Life cycle processes — Requirements engineering*
- [140] ISO/IEC/IEEE 42010:2011, *Systems and software engineering — Architecture description*
- [141] Project Management Institute (PMI) and IEEE Computer Society. *Software Extension to the PMBOK® Guide Fifth Edition*

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Standards Documents."

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

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

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the [IEEE-SA Website](#) at <http://ieeexplore.ieee.org/xpl/standards.jsp> or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE's standards development process, visit the IEEE-SA Website at <http://standards.ieee.org>.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: <http://standards.ieee.org/findstds/errata/index.html>. Users are encouraged to check this URL for errata periodically.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at <http://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, 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 the IEEE Standards Association.

Participants

The list of IEEE participants can be accessed at the following URL: http://standards.ieee.org/downloads/24765/24765-2017/24765-2017_wg-participants.pdf.

Abstract: This document provides a common vocabulary applicable to all systems and software engineering work. It was prepared to collect and standardize terminology. This document 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 and liaison organizations IEEE Computer Society and Project Management Institute. This document includes references to the active source standards for definitions so that systems and software engineering concepts and requirements can be further explored.

Keywords: computer, dictionary, information technology, software engineering, systems engineering, 24765

ICS 35.080; 01.040.35

ISBN 978-1-5044-4118-6 STD 22642 (PDF); 978-1-5044-4119-3 STDPD 22642 (Print)

Price group H

© ISO/IEC 2017 – All rights reserved
© IEEE 2017 – All rights reserved