INTERNATIONAL STANDARD

ISO/IEC/ IEEE 24765

First edition
2010-12-15

Systems and software engineering — Vocabulary

Ingénierie des systèmes et du logiciel — Vocabulaire

Reference number
ISO/IEC/IEEE 24765:2010(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York • NY 10016-5997, USA
E-mail stds.ipr@ieee.org
Web www.ieee.org

Published by ISO in 2011

Published in Switzerland

Contents

Page

Foreword ..... **v**

Introduction ..... **vi**

1 Scope **1**
1.1 Relationship of the print and internet-accessible versions **1**
1.2 Vocabulary structure **1**
1.3 PMI Glossary provisions **2**

2 Conformance **2**

3 Terms and definitions **2**

Annex A (informative) **List of Source Standards** **404**

Annex B (informative) **List of References** **409**

List of Figures

Page

Figure 1 — Activity 9
Figure 2 — Block Diagram 35
Figure 3 — Box diagram 38
Figure 4 — Bubble chart 39
Figure 5 — Call graph 42
Figure 6 — Case construct 45
Figure 7 — Categorization of software 46
Figure 8 — Data flow diagram 90
Figure 9 — Data structure diagram 92
Figure 10 — Directed graph 110
Figure 11 — Documentation tree 113
Figure 12 — Flowchart 145
Figure 13 — Graph 158
Figure 14 — If-then-else construct 168
Figure 15 — Input-process-output chart 178
Figure 16 — Modification request 222
Figure 17 — Structure chart 349
Figure 18 — UNTIL 387
Figure 19 — Web site 399
Figure 20 — WHILE 400

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of ISO/IEC JTC 1 is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

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.

ISO/IEC/IEEE 24765 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*, in cooperation with the Software & Systems Engineering Standards Committee of the IEEE Computer Society of the IEEE, 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 — Fourth Edition*, copyright 2008, Project Management Institute.

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 International Standard 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 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 International Standard 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 systems, rather than software, context.

This International Standard replaces IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology, which was contributed by the IEEE as a source document. The approach and lexical exactitude of IEEE Std 610.12-1990 served as a model for this International Standard. Nevertheless, approximately two-thirds of the definitions in this International Standard are new since IEEE Std 610.12 was last updated in 1990, a reflection of the continued evolution in the field.

The vocabulary is offered in both print and internet-accessible versions for ease of reference.

Systems and software engineering — Vocabulary

1 Scope

Consistent with ISO vocabulary standards, each technical committee is responsible for standard terminology in its area of specialization. This International Standard provides a common vocabulary applicable to all systems and software engineering work falling within the scope of ISO JTC 1/SC 7.

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;
- terms whose meaning in the information technology (IT) field could be directly inferred from their common English meaning.

1.1 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 24765 is issued periodically as a formal, published International Standard 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.

1.2 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.

An entry can consist of a single word, such as "software"; a phrase, such as "test case"; or an acronym, such as "CDR". Phrases are given in their natural order (test plan) rather than in reversed order (plan, test). Acronyms 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 International Standard 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, Fourth Edition. Sources are listed in Annex A. 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 illustrations taken from the source standards have been included to clarify selected definitions.

The following cross-references are used to show a term's relationship to other terms in the dictionary.

- *Syn* refers to a synonym: a term with the same meaning. Synonyms are listed under the preferred term and can be located by searching.

- cf. refers to related terms that are not synonyms.

1.3 PMI Glossary provisions

The Project Management Institute (PMI) Glossary definitions have been included without alteration in accordance with the copyright agreement. 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.

Many of the definitions from the PMI Glossary begin with a word or phrase in brackets, such as [Process], [Output/Input], [Technique]. These bracketed entries refer to the schema of the Project Management Institute, *A Guide to the Project Management Body of Knowledge (PMBOK®[1)] Guide)* – Fourth Edition, which provides further explanation.

2 Conformance

The definitions in this International Standard are drawn from normative standards and informative guidance documents, including ISO Technical Reports (TR). This International Standard may be used as a normative document for projects and organizations claiming conformance to the normative source standards. Where terms have multiple definitions, users should consult the source standards for further information on appropriate usage within a specific context. Annex B lists other references.

Terms, definitions, and notes use spelling preferred in the USA. 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*). Other correct spellings and capitalization of the terms, according to a national standard, an authoritative general dictionary or accepted style guide may be used with the definitions.

3 Terms and definitions

3.1
<Viewpoint> language
1. definitions of concepts and rules for the specification of an ODP system from the <viewpoint> viewpoint. ISO/IEC 10746-3:1996 Information technology — Open Distributed Processing — Reference Model: Architecture.4.2.1.1

NOTE Thus, engineering language is defined as "definitions of concepts and rules for the specification of an ODP system from the engineering viewpoint".

1) PMBOK is a trademark of the Project Management Institute, Inc. which is registered in the United States and other nations.

3.2
<X> federation
1. a community of <x> domains*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*5.1.2

3.3
<x> interceptor
1. an engineering object in a channel, placed at a boundary between <x> domains*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.11

NOTE 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.

3.4
1GL
1. first-generation language

cf. machine language

3.5
2GL
1. second-generation language

cf. assembly language

3.6
3GL
1. third-generation language

cf. high order language

3.7
4GL
1. fourth-generation language

3.8
5GL
1. fifth-generation language

3.9
A-0 context diagram
1. the only context diagram that is a required for a valid IDEF0 model, the A-0 diagram contains one box, which represents the top-level function being modeled, the inputs, controls, outputs, and mechanisms attached to this box, the full model name, the model name abbreviation, the model's purpose statement, and the model's viewpoint statement. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0*

3.10
A4, A5
1. International Standard paper sizes*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.1

NOTE A4 is 210 mm by 297 mm and A5 is 148 mm by 210 mm; see ISO 216:2007.

3.11
abend
1. abbreviation for abnormal end

3.12
abnormal end (abend)
1. termination of a process prior to completion

cf. abort, exception

3.13
abort
1. to terminate a process prior to completion

cf. abnormal end (abend), exception

3.14
absolute address
1. an 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. *Syn:* explicit address, specific address

cf. relative address, relocatable address, symbolic address, absolute assembler, absolute code, absolute instruction

3.15
absolute assembler
1. an assembler that produces absolute code

cf. relocating assembler

3.16
absolute code
1. code in which all addresses are absolute addresses. *Syn:* specific code

cf. relocatable code

3.17
absolute instruction
1. a computer instruction in which all addresses are absolute addresses

cf. direct instruction, effective instruction, immediate instruction, indirect instruction

3.18
absolute loader
1. a 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

3.19
abstract class
1. a class that cannot be instantiated independently. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.1

NOTE That is, instantiation must be accomplished via a subclass. A class for which every instance must also be an instance of a subclass in the cluster (a total cluster) is called an abstract class with respect to that cluster.

3.20
abstract data type
1. a 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

3.21
abstract design
1. a generic form that needs specialization (further design work) to produce concrete designs. ***2.*** design aimed at producing designs

3.22
abstraction
1. a view of an object that focuses on the information relevant to a particular purpose and ignores the remainder of the information. ***2.*** the process of formulating a view

cf. data abstraction

3.23
acceptability
1. the exposure to loss (financial or otherwise) that an organization is willing to tolerate from a risk

NOTE Risk acceptability may apply to an individual risk or to a collection of risks, such as the totality of risks confronting a project or enterprise. Acceptability may differ for different categories of risk and may depend on the cost of treatment or other factors.

3.24
acceptable
1. meeting stakeholder expectations that can be shown to be reasonable or merited. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.3.1

3.25
acceptance criteria
1. the criteria that a system or component must satisfy in order to be accepted by a user, customer, or other authorized entity. ***2.*** those criteria, including performance requirements and essential conditions, which must be met before project deliverables are accepted. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. requirement, test criteria

3.26
acceptance test
1. the 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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.07.

cf. acceptance testing, validation testing

3.27
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. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.1. *Syn:* validation testing, acceptance test. ***2.*** formal testing conducted to enable a user, customer, or other authorized entity to determine whether to accept a system or component. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.1.

cf. validation testing, acceptance test

3.28
access
1. to obtain the use of a resource. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.04

3.29
access facility
1. a 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

3.30
access method
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.08.03

3.31
access routine
1. a 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

3.32
access transparency
1. a distribution transparency which masks differences in data representation and invocation mechanisms to enable interworking between objects*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.1

3.33
accessibility
1. 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, Systems and software engineering--requirements for designers and developers of user documentation.*4.1

NOTE Although "accessibility" typically addresses users who have disabilities, the concept is not limited to disability issues.

3.34
accident
1. an unplanned event or series of events that results in death, injury, illness, environmental damage, or damage to or loss of equipment or property. *IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.1

3.35
accuracy
1. a qualitative assessment of correctness, or freedom from error. ***2.*** a quantitative measure of the magnitude of error

3.36
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 Accuracy is a qualitative concept. The term precision should not be used for "accuracy". [ISO/IEC Guide 99:2007 International vocabulary of metrology — Basic and general concepts and associated terms] A true value is a value consistent with the definition of a given particular quantity and this is a value that would be obtained by a perfect measurement. In contexts where perfect measurement is not practically feasible, a conventional true value is a value attributed to a particular quantity and accepted, sometimes by convention, as having an uncertainty appropriate for a given purpose. 'Conventional true value', in the same reference, is sometimes called assigned value, best estimate of the value, conventional value or reference value. The accuracy should be expressed in terms of the Mean magnitude of relative error.

3.37
ACID
1. Atomicity Consistency Isolation Durability*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.38
acquire project team
1. [Process] the process of confirming human resource availability and obtaining the team necessary to complete project assignments. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.39
acquirer
1. stakeholder that acquires or procures a product or service from a supplier*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.1. ***2.*** the individual or organization that specifies requirements for and accepts delivery of a new or modified software product and its documentation. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.1; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.1. *Syn:* buyer, customer, owner, purchaser

NOTE The acquirer may be internal or external to the supplier organization. Acquisition of a software product may involve, but does not necessarily require, a legal contract or a financial transaction between the acquirer and supplier.

3.40
acquisition
1. process of obtaining a system, software product or software service*. ISO/IEC 12207:2008 (IEEE Std 12207-2008) Systems and software engineering — Software life cycle processes.*4.2. ***2.*** process of obtaining a system product or service*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.2. *Syn:* outsourcing

3.41
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

3.42
action
1. element of a step that a user performs during a procedure*. ISO/IEC 26514, Systems and software engineering--requirements for designers and developers of user documentation.*4.2. ***2.*** a 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

3.43
action entry
1. an indication of the relevance of an action to a particular rule*. ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.9

3.44
action of interest
1. an action in a transaction which leads to a state change of significance to the transaction*. ISO/IEC 10746-3:1996 Information technology — Open Distributed Processing — Reference Model: Architecture.*13.7.1.2

3.45
action stub
1. a 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

3.46
activation
1. one occurrence of a function's transformation of some subset of its inputs into some subset of its outputs. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.3

3.47
activation constraint
1. a 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.4

3.48
active area
1. (on-screen documentation) area that responds to user input.4.3

EXAMPLE a window, icon or text field

3.49
active interconnection
1. a physical interaction mechanism allowing the action of one thing to cause a change or to stimulate an action in another thing. *IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.1

3.50
active redundancy
1. in fault tolerance, the use of redundant elements operating simultaneously to prevent, or permit recovery from, failures

cf. standby redundancy

3.51
active text
1. text displayed on the screen that responds to user input

3.52
active white space
1. area around textual or graphical elements, not including margins, which breaks up text, separates topic and subtopic groupings, indicates hierarchical and topical relationships, highlights information or makes text easier to read. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.54

3.53
activity
1. set of cohesive tasks of a process. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.3; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.3. ***2.*** a component of work performed during the course of a project. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***3.*** an 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.*** a defined body of work to be performed, including its required input information and output information. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*Annex E. ***5.*** collection of related tasks. *ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.1. ***6.*** element of work performed during the implementation of a process. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.2

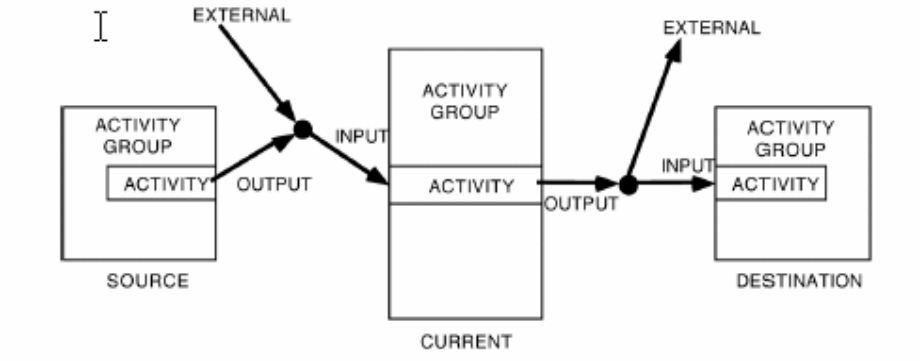


Figure 1 —Activity

NOTE An activity normally has an expected duration, cost, and resource requirements. Activities are often subdivided into tasks.

3.54
activity attributes
1. [Output/Input] 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) — Fourth Edition*

3.55
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) — Fourth Edition*

3.56
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) — Fourth Edition*

3.57
activity group
1. a set of related activities. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.* Annex E

3.58
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) — Fourth Edition*

3.59
activity list
1. [Output/Input] 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) — Fourth Edition*

3.60
activity type
1. a 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

3.61
actor
1. a role (with respect to that action) in which the enterprise object fulfilling the role participates in the action. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.3.1. ***2.*** organization or CASE tool that supplies and/or acquires SEE Services. *ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.4. ***3.*** in UML, someone or something outside the system that interacts with the system

NOTE It may be of interest to specify which actor initiates that action.

3.62
actual cost (AC)
1. total costs actually incurred and recorded in accomplishing work performed during a given time period for a schedule activity or work breakdown structure component. Actual cost can sometimes be direct labor hours alone, direct costs alone, or all costs including indirect costs. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* actual cost of work performed (ACWP)

cf. earned value management, earned value technique

3.63
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) — Fourth Edition*

3.64
ACWP
1. actual cost of work performed. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.65
adaptation data
1. data used to adapt a program to a given installation site or to given conditions in its operational environment

3.66
adaptation parameter
1. a 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

3.67
adapter
1. an object adapter. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.1

3.68
adaptive maintenance
1. modification of a software product, performed after delivery, to keep a software product usable in a changed or changing environment. *ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.1

EXAMPLE The operating system might be upgraded and some changes may be made to accommodate the new operating system.

NOTE Adaptive maintenance provides enhancements necessary to accommodate changes in the environment in which a software product must operate. These changes are those that must be made to keep pace with the changing environment.

3.69
added source statements
1. the count of source statements that were created specifically for the software product

3.70
address
1. a 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. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.3

3.71
address field
1. a field of a computer instruction that contains addresses, information necessary to derive addresses, or values of operands. *Syn:* address part

cf. operation field

3.72
address format
1. the number and arrangement of address fields in a computer instruction. ***2.*** the 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

3.73
address modification
1. an arithmetic, logical, or syntactic operation performed on an address

cf. effective address, indexed address, relative address, relocatable address

3.74
address space
1. the addresses that a computer program can access. ***2.*** the number of memory locations that a central processing unit can address

NOTE In some systems, this may be 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 may be accessible by other programs.

3.75
addressing exception
1. an 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

3.76
adjusted function point count (AFP)
1. the unadjusted function point count multiplied by the value adjustment factor. *ISO/IEC 20926:2003; Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE calculated using a specific formula for development project, enhancement project, and application; commonly called the function point count

3.77
adjusted size
1. a size based on the functional size multiplied by the technical complexity adjustment*. ISO/IEC 20968:2002; Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10

NOTE This measure does not represent functional size.

3.78
Administer Procurements
1. [Process] the process of managing procurement relationships, monitoring contract performance, and making changes and corrections as needed*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.79
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

3.80
ADT
1. Abstract Data Type*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.81
AE(I)
1. Application Entity (Invocation)*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.82
afferent
1. pertaining to a flow of data or control from a subordinate module to a superordinate module in a software system

cf. efferent

3.83
agent
1. an enterprise object that has been delegated (authority, responsibility, a function, etc) by and acts for another enterprise object (in exercising the authority, carrying out the responsibility, performing the function, etc)*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.5.7

NOTE An agent may be a party or may be the ODP system or one of its components. Another system in the environment of the ODP system may also be an agent. The delegation may have been direct, by a party, or indirect, by an agent of the party having authorization from the party to so delegate.

3.84
aggregate responsibility
1. a broadly stated responsibility that is eventually refined as specific properties and constraints*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.3

3.85
agreement
1. mutual acknowledgement of terms and conditions under which a working relationship is conducted*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.4; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.4

cf. contract

3.86
algebraic language
1. a 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

3.87
algorithm
1. a finite set of well-defined rules for the solution of a problem in a finite number of steps. ***2.*** a sequence of operations for performing a specific task. ***3.*** a finite ordered set of well-defined rules for the solution of a problem. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.05

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

3.88
algorithmic language
1. a programming language designed for expressing algorithms

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

EXAMPLE ALGOL

3.89
alias
1. an alternate name for an IDEF1X model construct (class, responsibility, entity, or domain). *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.4

3.90
allocated baseline
1. in configuration management, the initial approved specifications governing the development of configuration items that are part of a higher-level configuration item

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

3.91
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 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.

3.92
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

3.93
allocation
1. the process of distributing requirements, resources, or other entities among the components of a system or program. ***2.*** the result of the distribution of requirements, resources, or other entities among the components of a system or program. ***3.*** the decision to assign a function or decision to hardware, software, or humans. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.3

NOTE Allocation may be made entirely to hardware, software, or humans, or to some combination to be resolved upon further functional decomposition.

3.94
alpha testing
1. first stage of testing before a product is considered ready for commercial or operational use

cf. beta testing

NOTE often performed only by users within the organization developing the software

3.95
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.05

3.96
ALS
1. Application Layer Structure. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.97
alternate flow
1. the part of a use case that describes its alternative implementations. *Syn:* alternate path

NOTE It is also used to describe error conditions, since errors can be considered a kind of alternative.

3.98
alternate key
1. a candidate key of an entity other than the primary key. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.5

NOTE [key style]

3.99
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.06

3.100
analog computer
1. a computer whose operations are analogous to the behavior of another system and that accepts, processes, and produces analog data. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.05

EXAMPLE an abacus

3.101
analogous estimating
1. [Technique] an estimating technique that uses the values of parameters, such as scope, cost, budget, and duration or measures of scale such as size, weight, and complexity from a previous, similar activity as the basis for estimating the same parameter or measure for a future activity*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.102
analysis
1. the process of studying a system by partitioning the system into parts (functions, components, or objects) and determining how the parts relate to each other*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.1. ***2.*** investigation and collection phase of user documentation development that aims to specify types of users and their information needs.4.4

3.103
analysis model
1. algorithm or calculation combining one or more base and/or derived measures with associated decision criteria*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.2

3.104
analyst
1. a member of the technical community (such as a systems engineer or business analyst, developing the system requirements) who is skilled and trained to define problems and to analyze, develop, and express algorithms*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.1

3.105
ancestor (of a class)
1. a generic ancestor of the class or a parent of the class or an ancestor of a parent of the class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.6

cf. generic ancestor, reflexive ancestor

3.106
ancestral box
1. a box related to a specific diagram by a hierarchically consecutive sequence of one or more parent/child relationships*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.6

3.107
ancestral diagram
1. a diagram that contains an ancestral box*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.7

3.108
annotate
1. a 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

3.109
annotation
1. further documentation accompanying a requirement such as background information and/or descriptive material*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.2

3.110
announcement
1. an interaction - the 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture*.7.1.3

3.111
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 Std 1028-2008 IEEE Standard for Software Reviews and Audits*.3.1. ***2.*** anything observed in the documentation or operation of software or system that deviates from expectations based on previously verified software products, reference documents, or other sources of indicative behavior*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.4

3.112
ANSI
1. American National Standards Institute*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.113
anticipatory buffering
1. a buffering technique in which data are stored in a buffer in anticipation of a need for the data

cf. dynamic buffering, simple buffering

3.114
anticipatory paging
1. a 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

3.115
AP(I)
1. Application Process (Invocation)*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.116
aperiodic task
1. a task activated on demand. *Syn:* asynchronous task

3.117
API
1. Application Program Interface*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.118
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

3.119
application
1. a system for collecting, saving, processing, and presenting data by means of a computer. *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.*** a cohesive collection of automated procedures and data supporting a business objective. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual. Syn:* application system, information system

cf. system

NOTE It consists of one or more components, modules, or subsystems. Frequently a synonym for "system", the word "application" is preferred to express more precisely the nature of the subject matter of functional size measurement.

3.120
application administration function
1. functions performed by users which include installation, configuration, application backup, maintenance (patching and upgrading) and de-installation. *ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.1

3.121
application area
1. a general term for a grouping of applications that handle a specific business area. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** 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) — Fourth Edition*

NOTE It corresponds to an administrative level for management purposes.

3.122
application area level
1. the management level responsible for managing maintenance activities as well as new development or major enhancement projects for one or more applications. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.123
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* ***2.*** the border between the software being measured and the user. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.124
application engineering
1. the process of constructing or refining application systems by reusing assets. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.1

3.125
application frameworks
1. a subsystem design made up of a collection of abstract and concrete classes and interfaces between them

NOTE Frameworks are often instantiation of a number of patterns.

3.126
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.*** a count that provides a measure of the current functionality the application provides to the user. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE i.e., the functionality already provided to the user or that is still to be provided. With it, the effort required to support the realized application can also be determined.

3.127
application generator
1. a code generator that produces programs to solve one or more problems in a particular application area

EXAMPLE a payroll generator

3.128
application manager
1. a person responsible for managing projects and support activities for one or more application areas. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.129
application problem
1. a problem submitted by an end user and requiring information processing for its solution. *ISO/IEC 2382-20:1990 Information technology — Vocabulary — Part 20: System development.*20.01.13

3.130
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.4.5. ***2.*** software or a program that is specific to the solution of an application problem. *ISO/IEC 2382-1:1993 Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.01. ***3.*** software designed to fulfill specific needs of a user

3.131
application-oriented language
1. a computer language with facilities or notations applicable primarily to a single application area

cf. authoring language, query language, specification language

EXAMPLE a language for computer-assisted instruction or hardware design

3.132
apportioned effort
1. effort applied to project work in proportion to measurable discrete work efforts, but which is not readily divisible into discrete efforts

cf. discrete effort

3.133
appraisal findings
1. results of an appraisal that identify the most important issues, problems, or opportunities for process improvement within the appraisal scope

NOTE Appraisal findings are inferences drawn from corroborated objective evidence.

3.134
appraisal participants
1. members of the organizational unit who participate in providing information during an appraisal

3.135
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

3.136
approval
1. written notification, by an authorized representative, that an information item appears to satisfy requirements, is complete. *ISO/IEC 15289:2006, Systems and software engineering — Contents of systems and software life cycle information products (Documentation)* 5.1

NOTE Such approval does not shift responsibility from the supplier to meet requirements under a two-party situation.

3.137
approved change request
1. [Output/Input] 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) — Fourth Edition*

3.138
approved modification
1. the 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 There may be a many-to-many relationship of "proposed change" to "approved modification". A proposed change may cause modifications in several SCIs (even if only to the code and its test case). A modification may originate from several proposed changes, approved simultaneously or over a period of time while the modification is still in progress.

3.139
A-profile
1. Application profile. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.140
arc
1. a directed edge of a net which may 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

3.141
arc annotation
1. an expression that may 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. The expression must evaluate to a multiset over the type of the arc's associated place

3.142
architect
1. the person, team, or organization responsible for systems architecture. *IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*3.2

3.143
architecting
1. the activities of defining, documenting, maintaining, improving, and certifying proper implementation of an architecture. *IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*3.3

3.144
architectural description (AD)
1. a collection of products to document an architecture*. IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*3.4

3.145
architectural design
1. the 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

3.146
architectural design phase
1. the 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

3.147
architectural design review

1. a joint acquirer-supplier review to evaluate the technical adequacies of the software architectural design as depicted in the software design descriptions

3.148
architectural structure
1. a 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

3.149
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

3.150
architecture
1. fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.5. ***2.*** the organizational structure of a system or component. ***3.*** the organizational structure of a system and its implementation guidelines. *Syn:* architectural structure

cf. component, module, subprogram, routine

NOTE sometimes refers to the design of a system's hardware and software components

3.151
archival pages
1. on-line data that is 1) not expected to change, and 2) no longer maintained*. IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle*.3.1.1

NOTE This data also may not be readily renderable by future tools.

3.152
argument
1. an independent variable. ***2.*** a specific value of an independent variable. ***3.*** a 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 $E = mc^2$

3.153
argument sort
1. the 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. *Syn:* input sort

3.154
arity
1. the number of roles that participate in a relationship. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2. ***2.*** the 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 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.

3.155
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

3.156
arrow
1. a 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 Std 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

3.157
arrow label
1. a noun or noun phrase associated with an arrow segment to signify the arrow meaning of the arrow segment. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.9

NOTE Specifically, an arrow label identifies the object type set that is represented by an arrow segment.

3.158
arrow meaning
1. the object types of an object type set, regardless of how these object types may be collected, aggregated, grouped, bundled, or otherwise joined within the object type set. *IEEE Std 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

3.159
arrow role
1. the 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.12

NOTE There are four arrow roles: input, control, output, and mechanism.

3.160
arrow segment
1. a 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.13

3.161
artificial intelligence (AI)
1. the 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.12

3.162
artificial language
1. a language whose rules are explicitly established prior to its use. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.09. *Syn:* formal language

3.163
ask
1. the 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

3.164
ASO
1. Application Service Object. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.165
assemble
1. to translate a computer program expressed in an assembly language into its machine language equivalent.
2. the process of constructing from parts one or more identified pieces of software. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.2

cf. compile, disassemble, interpret

3.166
assemble-and-go
1. an operating technique in which there are no stops between the assembling, linking, loading, and execution of a computer program

3.167
assembled origin
1. the 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

3.168
assembler
1. a computer program that translates programs expressed in assembly language into their machine language equivalents

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

3.169
assembly code
1. computer instructions and data definitions expressed in a form that can be recognized and processed by an assembler. *Syn:* assembler code

cf. compiler code, interpretive code, machine code

3.170
assembly language
1. a 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. *Syn:* assembler language, low-level language, second-generation language

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

3.171
assertion
1. a 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.*** a 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 Types include input assertion, loop assertion, output assertion.

3.172
assessed capability
1. the output of one or more relevant process assessments conducted in accordance with the provisions of ISO/IEC 15504. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.2

3.173
assessment
1. an 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

3.174
assessment constraints
1. restrictions placed on the use of the assessment outputs and on the assessment team's freedom of choice regarding the conduct of the assessment. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.3

3.175
assessment indicator
1. sources of objective evidence used to support the assessors' judgment in rating process attributes. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.4

EXAMPLE work products, practice, or resource

3.176
assessment input
1. information required before a process assessment can commence. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.5

3.177
assessment instrument
1. a tool or set of tools that is used throughout an assessment to assist the assessor in evaluating the performance or capability of processes, in handling assessment data and in recording the assessment results. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.6

3.178
assessment output
1. the tangible results from an assessment. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.7

cf. assessment record

3.179
assessment participant
1. an individual who has responsibilities within the scope of the assessment. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.8

EXAMPLE the assessment sponsor, assessors, and organizational unit members

3.180
assessment process
1. a determination of the extent to which the organization's standard processes contribute to the achievement of its business goals and to help the organization focus on the need for continuous process improvement [. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.9

3.181
assessment purpose
1. a statement, provided as part of the assessment input, which defines the reasons for performing the assessment. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.10

3.182
assessment record
1. an 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 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.11

3.183
assessment scope
1. a definition of the boundaries of the assessment, provided as part of the assessment input, encompassing the organizational limits of the assessment, the processes to be included, and the context within which the processes operate. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.12

cf. process context

3.184
assessment sponsor
1. the 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 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.13

3.185
assessment team
1. one or more individuals who jointly perform a process assessment. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.14

3.186
assessor
1. an individual who participates in the rating of process attributes*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.15

NOTE An assessor is either a competent assessor or a provisional assessor.

3.187
asset
1. an item that has been designed for use in multiple contexts*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.3. ***2.*** an advantage or resource*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***3.*** a capital asset of the enterprise*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.188
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. *Syn:* binding

3.189
assignment statement
1. a computer program statement that assigns a value to a variable

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

EXAMPLE Y = X - 5

3.190
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

3.191
assistive technologies
1. hardware or software that is added to or incorporated within a system that increases accessibility for an individual EXAMPLES Braille displays, screen readers, screen magnification software and eye tracking devices are assistive technologies*. ISO/IEC 25062:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports.*4.11

3.192
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.*** a 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

3.193
association management facility
1. a 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

3.194
associative class
1. a class introduced to resolve a many-to-many relationship*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.7

3.195
associative entity
1. an entity used to represent a relationship between other entities An associative entity is used when a relationship does not otherwise provide sufficient mechanisms*. ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.4.2

3.196
associative entity type
1. an entity type that contains attributes which further describe a many-to-many relationship between two other entity types*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. entity type

3.197
associative literal
1. a literal that denotes an instance in terms of its value*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.8

NOTE The form of expression used to state an associative literal is className with propertyName: propertyValue.

3.198
assumptions
1. factors that, for planning purposes, are considered to be true, real, or certain without proof or demonstration*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.199
assumptions analysis
1. [Technique] 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) — Fourth Edition*

3.200
asynchronous
1. pertaining to two or more processes that do not depend upon the occurrence of specific events such as common timing*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms*.01.01.29

3.201
asynchronous I/O device
1. an I/O device that generates an interrupt after producing some input or generating some output

3.202
asynchronous I/O device interface task
1. a task that interfaces to an I/O device and is activated by interrupts from that device

3.203
asynchronous message communication
1. communication in which a producer task sends a message to a consumer task and does not wait for a response. *Syn:* loosely coupled message communication

NOTE A message queue could build up between the tasks.

3.204
atomic type
1. a data type, each of whose members consists of a single, nondecomposable data item. *Syn:* primitive type

cf. composite type

3.205
attribute
1. a property associated with a set of real or abstract things that is some characteristic of interest. *IEEE Std 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 15939:2007, Systems and software engineering--Measurement process*.3.2; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*.4.3. ***3.*** a measurable physical or abstract property of an entity. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*.2.1. ***4.*** an identifiable association between an object and a value. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.2.2. ***5.*** a function from the instances of a class to the instances of the value class of the attribute. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.9. ***6.*** a unique item of information about an entity. *ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10. ***7.*** a single-valued characteristic of an entity or relationship. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.4.2

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

NOTE can refer either to general characteristics such as reliability, maintainability, and usability or to specific features of a software product. ISO 9000 distinguishes two types of attributes: a permanent characteristic existing inherently in something; and an assigned characteristic of a product, process or system (e.g. the price of a product, the owner of a product). The assigned characteristic is not an inherent quality characteristic of that product, process or system. 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 may simply be the name of the value class (as long as using the value class name does not cause ambiguity).

3.206
attribute for quality measure
1. attribute that relates to software product itself, to the use of the software product or to its development process. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*.4.4

NOTE Attributes for quality measure are used in order to obtain quality measure elements.

3.207
attribute indicator
1. an assessment indicator that supports the judgment of the extent of achievement of a specific process attribute. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.16

3.208
attribute name
1. a role name for the value class of the attribute. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.10

3.209
attributed relationship
1. a relationship that has attributes. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.4.2

3.210
attributive entity type
1. an entity type that further describes one or more attributes of another entity type*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. entity

3.211
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.6

NOTE There may be different audiences for documentation (for example, management, data entry, maintenance) that determine the content, structure, and use of the intended documentation.

3.212
audience research
1. planned process of interview, and of the analysis of interview records and personnel records*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.4

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

3.213
audit
1. independent assessment of products and processes, conducted by an authorized person to assess compliance with requirements*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.5. ***2.*** an 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 Std 1028-2008 IEEE Standard for Software Reviews and Audits.*3.2. ***3.*** an independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria ***4.*** 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 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.6

NOTE An audit should result in a clear indication of whether the audit criteria have been met.

3.214
authoring language
1. a high-level programming language used to develop courseware for computer-assisted instruction

cf. authoring system

3.215
authoring system
1. a programming system that incorporates an authoring language

3.216
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) — Fourth Edition*

3.217
authorization
1. a prescription that a particular behavior shall not be prevented*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.4.2

NOTE Unlike a permission, an authorization is an empowerment.

3.218
automate
1. to make a process or equipment automatic*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.13

3.219
automated or assisted software process
1. software process that is performed either fully or partially supported by CASE tools*. ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.3

3.220
automated verification system
1. a 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.*** a software tool that automates part or all of the verification process

3.221
automatic
1. pertaining to a process or equipment that, under specified conditions, functions without human intervention*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.12

3.222
automation
1. the conversion of processes or equipment to automatic operation, or the results of the conversion*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.14

3.223
availability
1. the degree to which a system or component is operational and accessible when required for use. ***2.*** ability of a component or service to perform its required function at a stated instant or over a stated period of time*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.1

cf. error tolerance, fault tolerance, robustness

NOTE Often expressed as a probability. Availability is usually expressed as a ratio of the time that the service is actually available for use by the business to the agreed service hours.

3.224
B5
1. International Standard paper size, 176 mm by 250 mm*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.5

NOTE See ISO 216:2007.

3.225
BAC
1. budget at completion*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.226
back matter
1. material that appears at the end of a book or manual, such as an index*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.6

3.227
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

3.228
background processing
1. the 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

3.229
backout
1. to undo the effects of a commit

NOTE Often by introducing a new commit that restores things to their previous state.

3.230
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

cf. mutation testing

3.231
backup
1. a 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. *Syn:* back-up

3.232
backup programmer
1. the assistant leader of a chief programmer team

cf. chief programmer

NOTE 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.

3.233
backward pass
1. the calculation of late finish dates and late start dates for the uncompleted portions of all schedule activities. Determined by working backwards through the schedule network logic from the project's end date*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. schedule network analysis

3.234
backward recovery
1. the reconstruction of a file to a given state by reversing all changes made to the file since it was in that state. ***2.*** a 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

3.235
bag
1. a collection class whose members are unordered but in which duplicates are meaningful*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.11

cf. list, set

3.236
base address
1. an 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

3.237
base functional component (BFC)
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. *Syn:* functional service

EXAMPLE a functional user requirement could be "Maintain Customers" which may consist of the following BFCs: "Add a new customer", "Report Customer Purchases", and "Change Customer Details". Another example might include a collection of logically related business data maintained by the software under study such as "Customer Details"

3.238
base measure
1. measure defined in terms of an attribute and the method for quantifying it*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.3; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.5

NOTE A base measure is functionally independent of other measures.

3.239
base practice
1. an activity that, when consistently performed, contributes to achieving a specific process purpose*. ISO/IEC 15504-1:2004 Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.17

3.240
baseline
1. 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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.6, *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.7. ***2.*** 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 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3.1. ***3.*** agreement or result designated and fixed at a given time, from which changes require justification and approval. ***4.*** document or a set of such documents formally designated and fixed at a specific time during the life cycle of a configuration item ***5.*** work product that has been placed under formal configuration management. ***6.*** snapshot of the state of a service or individual configuration items at a point in time*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.2. ***7.*** description of a system and its components (configuration items) at a particular period including any approved updates. ***8.*** an approved plan (for a project), plus or minus approved changes. It is compared to actual performance to determine if performance is within acceptable variance thresholds. Generally refers to the current baseline, but may refer to the original or some other baseline. Usually used with a modifier (e.g., cost performance baseline, schedule baseline, performance measurement baseline, technical baseline)*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

NOTE A baseline should be changed only through formal configuration management procedures. Some baselines may be project deliverables while others provide the basis for further work. Baselines, plus approved changes from those baselines, constitute the current configuration identification.

3.241
baseline design
1. a system design that has been agreed on by all stakeholders interested in the system development

3.242
baseline document
1. a system/software document that defines a work product that has been placed under configuration management

EXAMPLE system specifications, requirements specifications, and design specifications

3.243
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

3.244
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

3.245
basic engineering object
1. an engineering object that requires the support of a distributed infrastructure. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.1

3.246
basic flow
1. the part of a use case that describes its most common implementation. *Syn:* basic path, happy day scenario

NOTE The basic flow is written assuming that no errors or alternatives exist.

3.247
basic interworking facility
1. a set of service primitives which have a direct correspondence with computational signals which model computational operations. *ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*3.3.4

3.248
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

3.249
basis set
1. the 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

3.250
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

3.251
bathtub curve
1. a graph of the number of failures in a system or component as a function of time

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

3.252
BCWP
1. budgeted cost of work performed. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.253
BCWS
1. budgeted cost of work scheduled. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.254
behavior
1. observable activity of a system, measurable in terms of quantifiable effects on the environment whether arising from internal or external stimulus. *IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description.*3.1. ***2.*** the peculiar reaction of a thing under given circumstances. *IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description.*3.1. ***3.*** the aspect of an instance's specification that is determined by the state-changing operations it can perform. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.12

3.255
behavior specification
1. a structured collection of data that describes the potential variety of behavior possible from a system. *IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections — Classification and Description.*3.2

3.256
benchmark
1. a standard against which measurements or comparisons can be made. ***2.*** a procedure, problem, or test that can be used to compare systems or components to each other or to a standard. ***3.*** a recovery file

3.257
BEO
1. Basic Engineering Object. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.258
beta testing
1. second stage of testing when a product is in limited production use

cf. alpha testing

NOTE often performed by a client or customer

3.259
BFC
1. base functional component. *ISO/IEC 14143-1:2007, Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts.*4

3.260
BFC class
1. defined group of BFC types. *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.1

3.261
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'

3.262
bidirectional traceability
1. association among two or more logical entities that is discernable in either direction (to and from an entity)

cf. requirements traceability

3.263
big-bang testing
1. a 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

3.264
bill of materials (BOM)
1. documented formal hierarchical tabulation of the physical assemblies, subassemblies, and components needed to fabricate a product

3.265
binary digit (bit)
1. a unit of information that can be represented by either a zero or a one. ***2.*** an element of computer storage that can hold a unit of information as in (1). ***3.*** a numeral used to represent one of the two digits in the binary numeration system; zero (0) or one (1)

3.266
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

3.267
binder
1. an engineering object in a channel, which maintains a distributed binding between interacting basic engineering objects*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.10

3.268
binding endpoint identifier
1. an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.15

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

NOTE The same form of binding endpoint identifier can be used, whether the binding involved is either local or distributed.

3.269
binding object
1. a computational object which supports a binding between a set of other computational objects. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.14

3.270
bit
1. either of the digits 0 or 1 when used in the binary numeration system. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.08. ***2.*** binary digit

3.271
bit steering
1. a microprogramming technique in which the meaning of a field in a microinstruction is dependent on the value of another field in the microinstruction. *Syn:* immediate control

cf. residual control, two-level encoding

3.272
black box
1. a 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

3.273
block
1. a 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

3.274
block diagram
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.27. ***2.*** a 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. *Syn:* configuration diagram, system resources chart

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

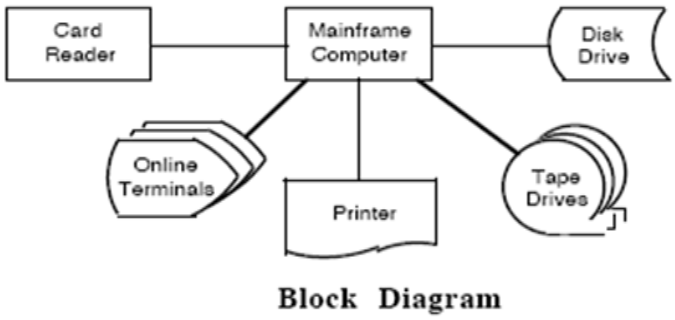


Figure 2 — Block Diagram

3.275
blocking factor
1. the number of records, words, characters, or bits in a block

3.276
block-structured language
1. a 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

3.277
BMT
1. Bench Mark Test. *ISO/IEC 14102:2008, Information technology — Guideline for the evaluation and selection of CASE tools.*8.2

3.278
body metadata
1. elements in the body of an HTML document providing administrative and/or navigational facilities for the user or administrator. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.2

3.279
BOM
1. bill of materials

3.280
Boolean expression
1. an 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

3.281
Boolean signature
1. a 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

3.282
boot
1. to initialize a computer system by clearing memory and reloading the operating system

NOTE derived from bootstrap

3.283
bootstrap
1. a 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

3.284
bootstrap loader
1. a short computer program used to load a bootstrap

3.285
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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.11

cf. topdown. critical piece first

EXAMPLE bottom-up design, bottom-up testing

3.286
bottom-up design
1. a design approach in which low-level pieces of a system are combined into an overall design. ***2.*** the 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

3.287
bottom-up estimating
1. [Technique] a method of estimating a component of work. The work is decomposed into more detail. An estimate is prepared of what is needed to meet the requirements of each of the lower, more detailed pieces of work, and these estimates are then aggregated into a total quantity for the component of work. The accuracy of bottom-up estimating is driven by the size and complexity of the work identified at the lower levels*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.288
boundary
1. conceptual interface between the software under study and its users*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.2

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

3.289
boundary arrow
1. an arrow with one end (source or use) not connected to any box in a diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.14

cf. internal arrow

3.290
boundary ICOM code
1. an 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.15

3.291
boundary value
1. a data value that corresponds to a minimum or maximum input, internal, or output value specified for a system or component

cf. stress testing

3.292
box
1. a rectangle containing a box name, a box number, and possibly a box detail reference and representing a function in a diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.16

3.293
box detail reference
1. a square enclosure encompassing a box number, which indicates that the box is decomposed or detailed by a child diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.17

3.294
box diagram
1. a control flow diagram consisting of a rectangle that is subdivided to show sequential steps, if-then-else conditions, repetition, and case conditions. *Syn:* Chapin chart, Nassi-Shneiderman chart

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

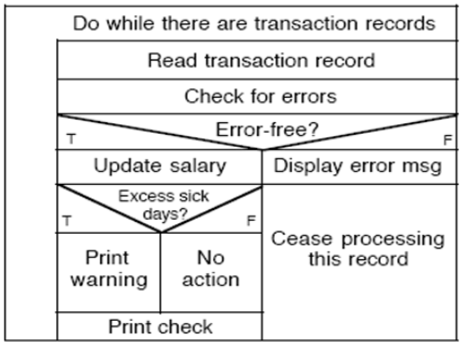


Figure 3 — Box diagram

3.295
box ICOM code
1. an ICOM code that maps a tunneled boundary arrow to an arrow attached to some ancestral box. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.18

3.296
box name
1. the verb or verb phrase placed inside a box that names the modeled function. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.19

cf. function name

NOTE A box takes as its box name the function name of the function represented by the box.

3.297
box number
1. a single digit (0, 1, 2, ..., 9) placed in the lower right corner of a box to uniquely identify that box in a diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.20

NOTE The only box that may be numbered 0 is the box that represents the A0 function in A-0 and A-1 context diagrams.

3.298
brainstorming
1. [Technique] 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) — Fourth Edition*

3.299
branch
1. a computer program construct in which one of two or more alternative sets of program statements is selected for execution. ***2.*** a point in a computer program at which one of two or more alternative sets of program statements is selected for execution. ***3.*** a junction at which a root arrow segment (going from source to use) divides into two or more arrow segments. *IEEE Std 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.*** a set of evolving source file versions

cf. case, jump, go to, if-then-else

NOTE Every branch is identified by a tag. Often, a branch identifies the file versions that have been or will be released as a product release. May 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.

3.300
branch testing
1. testing designed to execute each outcome of each decision point in a computer program

cf. path testing, statement testing

3.301
breakpoint
1. a point in a computer program at which execution can be suspended to permit manual or automated monitoring of program performance or results

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

3.302
bubble chart
1. a 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 4 — Bubble chart

3.303
budget
1. a planned sequence of expenditures over time with monetary costs assigned to specific tasks or jobs. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** 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) — Fourth Edition*

cf. estimate

NOTE often used also to refer to work effort as well as, or instead of, money

3.304
budget at completion (BAC)
1. the sum of all the budgets established for the work to be performed on a project or a work breakdown structure component or a schedule activity. The total planned value for the project. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.305
buffer
1. a 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.*** a routine that accomplishes the objectives in (1). ***3.*** to allocate, schedule, or use devices or storage areas as in (1)

3.306
build
1. an operational version of a system or component that incorporates a specified subset of the capabilities that the final product will provide

3.307
built-in class
1. a class that is a primitive in the IDEF1X metamodel. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.13

3.308
bundle
1. an 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.22. ***2.*** 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.22

3.309
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

3.310
busy
1. pertaining to a system or component that is operational, in service, and in use

cf. down, idle, up

3.311
busy time
1. in computer performance engineering, the period of time during which a system or component is

cf. down time, idle time, set-up time, up time

NOTE operational, in service, and in use

3.312
buyer
1. the acquirer of products, services, or results for an organization. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** an individual or organization responsible for acquiring a product or service for use by themselves or other users. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2. ***3.*** the person or organization that accepts the system and pays for the project. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2

cf. customer

NOTE A software system could be such a product or service.

3.313
byte
1. a 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.*** an element of computer storage that can hold a group of bits as in (1). ***3.*** a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.09

3.314
CAD
1. Computer Aided Design. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.315
calculator
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.17

NOTE A calculator performs some of the functions of a computer, but usually operates only with frequent human intervention.

3.316
calendar unit
1. the smallest unit of time used in scheduling a project. Calendar units are generally in hours, days, or weeks, but can also be in quarter years, months, shifts, or even in minutes. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.317
call
1. a transfer of control from one software module to another, usually with the implication that control will be returned to the calling module ***2.*** a 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

cf. go to

3.318
call arrow
1. an arrow that enables the sharing of detail between IDEF0 models (linking them together) or within an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.23

NOTE 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.

3.319
call by name
1. a 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 Because the expression is evaluated each time its corresponding formal parameter is used in the called module, the value of the parameter may change during the execution of the called module.

3.320
call by reference
1. a method for passing parameters, in which the calling module provides to the called module the address of the parameter to be passed. *Syn:* call by address, call by location

cf. call by name, call by value

NOTE With this method, the called module has the ability to change the value of the parameter stored by the calling module.

3.321
call by value
1. a 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 With this method, the called module cannot change the value of the parameter as stored by the calling module.

3.322
call graph
1. a diagram that identifies the modules in a system or computer program and shows which modules call one another. *Syn:* call tree, tier chart

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

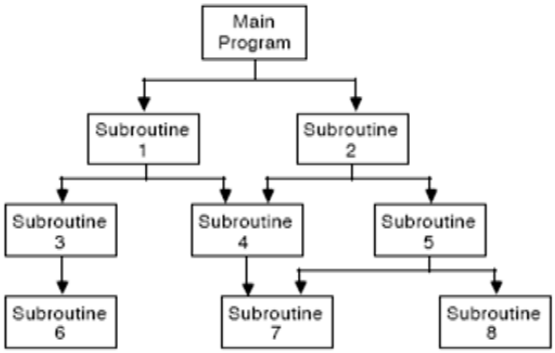


Figure 5 — Call graph

NOTE The result is not necessarily the same as that shown in a structure chart.

3.323
call list
1. the ordered list of arguments used in a call to a software module

3.324
call reference
1. a page reference attached to a call arrow*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.24

3.325
called diagram
1. a decomposition diagram invoked by a calling box and identified by a page reference attached to a call arrow*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.25

3.326
calling box
1. box that is detailed by a decomposition diagram that is not the box's child diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.26

NOTE A call arrow is attached to the bottom of a calling box.

3.327
calling sequence
1. a sequence of computer instructions and, possibly, data necessary to perform a call to another module

3.328
camera-ready originals
1. set of images on paper, photographic film or another suitable medium from which a printing plate can be made by direct photographic transfer, and where each image contains all of the necessary text and graphic elements for one complete page of paper documentation, with each element in the correct position*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.7

3.329
candidate FSM method
1. documented software size measurement method submitted for conformity evaluation*. ISO/IEC 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.1

3.330
candidate key
1. an attribute, or combination of attributes, of an entity for which no two instances agree on the values*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.14

NOTE [key style]

3.331
capability dimension
1. the set of elements in a process assessment model explicitly related to the measurement framework for process capability*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.18

NOTE The attributes are organized into capability levels, comprising an ordinal scale of process capability.

3.332
capability indicator
1. an assessment indicator that supports the judgment of the process capability of a specific process*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.19

NOTE An attribute indicator is a specific instance of a capability indicator.

3.333
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

3.334
capable process
1. process that can satisfy specified product quality, service quality, and process-performance objectives

cf. stable process, standard process, statistically managed process

3.335
capital expenditure
1. a form of spending in which an enterprise trades money (capital) for acquisition of tangible objects such as furniture, computers, and the like*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.336
capsule
1. a configuration of engineering objects forming a single unit for the purpose of encapsulation of processing and storage*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.4

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

3.337
capsule manager
1. an engineering object which manages the engineering objects in a capsule*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture*.8.1.5

3.338
cardinality
1. the 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.*** a specification of how many instances of a first class may or must exist for each instance of a second (not necessarily distinct) class, and how many instances of a second class may or must exist for each instance of a first class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.15

cf. cardinality constraint

NOTE For each direction of a relationship, the cardinality can be constrained.

3.339
cardinality constraint
1. a constraint that limits the number of instances that can be associated with each other in a relationship*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.16. ***2.*** a constraint that limits the number of members in a collection*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.16

cf. cardinality

3.340
carrier
1. a 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

3.341
case
1. a 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. *Syn:* multiple exclusive selective construct

cf. go to, jump, if-then-else. multiple inclusive selective construct

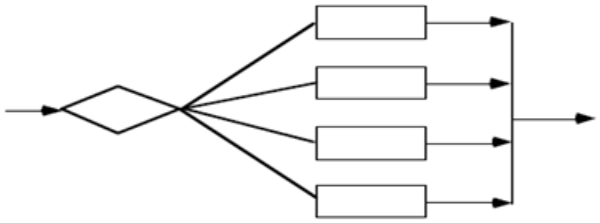


Figure 6 — Case construct

3.342
CASE needs
1. organizational requirements which are met by CASE tool characteristics*. ISO/IEC TR 14471:2007, Information technology — Software engineering — Guidelines for the adoption of CASE tools.*2.1.3

NOTE These characteristics are detailed in ISO/IEC 14102:1995. They include management process, development process, maintenance, documentation, configuration management, quality assurance, verification, validation, environment needs, CASE tool integrability, quality characteristics, acquisition needs, implementation needs, support indicators, and certification requirements.

3.343
CASE tool
1. software tool used for computer-aided software engineering (CASE)*. IEEE Std 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

NOTE: A CASE tool may provide support in only selected functional areas or in a wide variety of functional areas

3.344
cast
1. to treat an object of one type as an object of another type*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.17

cf. coerce

3.345
catastrophic failure
1. a failure of critical software

3.346
categorization scheme
1. an orderly combination of views and categories related to software*. ISO/IEC TR 12182:1998, Information technology — Categorization of software.*4.1

3.347
category
1. a specifically defined division or grouping of software based upon one or more attributes or characteristics*. ISO/IEC TR 12182:1998, Information technology — Categorization of software.*4.3. ***2.*** an attribute of an anomaly to which a group of classifications belongs*. IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies.*3.2

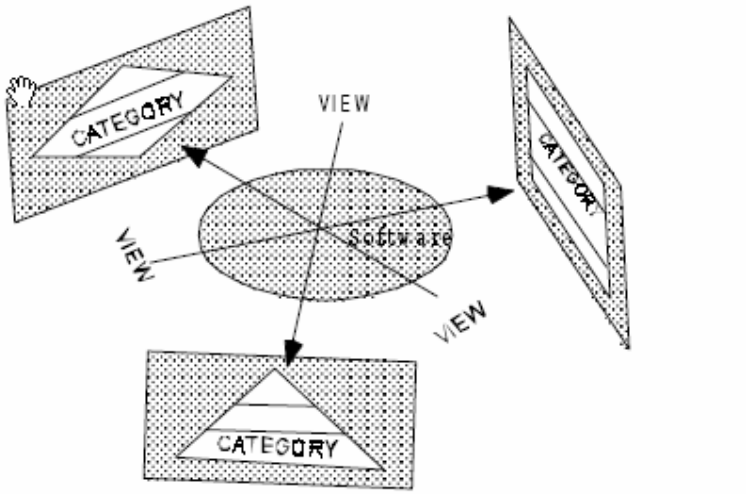


Figure 7 — Categorization of software

3.348
category entity
1. an entity whose instances represent a subtype or subclassification of another entity (generic entity)*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.21. *Syn:* subclass, subtype

NOTE: [key style]

3.349
causal analysis
1. analysis of a defect to determine its cause

3.350
caution
1. Advisory in software user documentation that performing some action may lead to consequences that are unwanted or undefined, such as loss of data or an equipment problem*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.7

cf. note, warning

3.351
CCB
1. configuration control board. ***2.*** change control board

3.352
CCCS
1. Client Conversion Code Sets. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.3

3.353
CCS
1. Conversion Code Sets. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.3

3.354
CD
1. Compact Disk. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.355
CDIF
1. CASE Data Interchange Format (originally). *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*5.2

3.356
CDIF clear text encoding
1. a clear text file encoding of a CDIF transfer file. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

3.357
CDIF exporter
1. a tool that creates a CDIF transfer file. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

3.358
CDIF family of standards
1. the CDIF family of standards is composed of a 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

3.359
CDIF graphical notation
1. the 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

3.360
CDIF identifier
1. an 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

3.361
CDIF importer
1. a 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

3.362
CDIF metaidentifier
1. a 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

3.363
CDIF meta-metamodel
1. the description of the set of concepts and notations used to define a metamodel. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

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

3.364
CDIF semantic metamodel
1. the description of the set of concepts and notations used to define a model. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

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

3.365
CDIF transfer
1. a combination of a particular syntax, a particular encoding of that syntax, and a metamodel. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE In other words, a complete definition of the format and contents of a transfer

3.366
CDIF transfer file
1. a transfer file conforming to ISO/IEC 15475. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.1

3.367
CDIF transfer format
1. a 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

3.368
CDIF transfer syntax and encoding
1. a standard vehicle format supported by CDIF. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE The combination of SYNTAX.1 and ENCODING.1 forms the initial CDIF transfer syntax and encoding.

3.369
CDR
1. critical design review. ***2.*** common data representation. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.370
central processing unit (CPU)
1. a functional unit that consists of one or more processors and their internal storage. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.01

3.371
certification
1. a written guarantee that a system or component complies with its specified requirements and is acceptable for operational use ***2.*** a formal demonstration that a system or component complies with its specified requirements and is acceptable for operational use ***3.*** the process of confirming that a system or component complies with its specified requirements and is acceptable for operational use

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

3.372
certification artifact
1. the tangible results from a certification process

EXAMPLE inspection checklists, metrics, problem reports

3.373
certification criteria
1. a set of standards, rules, or properties to which an asset must conform in order to be certified to a certain level

NOTE Certification criteria are defined by a certification policy. Certification criteria may be specified as a set of certification properties that must be met.

3.374
certification process
1. the process of assessing whether an asset conforms to predetermined certification criteria appropriate for that class of asset

3.375
certification property
1. a statement about some feature or characteristic of an asset that may be assessed as being true or false during a certification process

NOTE Properties may 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.

3.376
Cfsu
1. COSMIC functional size unit*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.* 4

3.377
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

3.378
chain type
1. a classification of chains which is defined by the sequence of tasks types*. ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.4

NOTE The emulated users submit only chains of specified chain types to the SUT.

3.379
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. *Syn:* enhancement

3.380
change authority
1. configuration board*. ISO/IEC TR 15846:1998, Information technology — Software life cycle processes — Configuration Management.*4.2

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

3.381
change control
1. identifying, documenting, approving or rejecting, and controlling changes to the project baselines*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. version control

3.382
change control board (CCB)
1. a formally constituted group of stakeholders responsible for reviewing, evaluating, approving, delaying, or rejecting changes to a project, with all decisions and recommendations being recorded*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.383
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.8

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

3.384
change control system
1. [Tool] a collection of formal documented procedures that define how project deliverables and documentation will be controlled, changed, and approved. In most application areas, the change control system is a subset of the configuration management system*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.385
change dump
1. a selective dump of those storage locations whose contents have changed since some specified time or event. *Syn:* differential dump

cf. dynamic dump, memory dump, postmortem dump, selective dump, snapshot dump, static dump

3.386
change management
1. judicious use of means to effect a change, or a proposed change, to a product or service

cf. configuration management

3.387
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

3.388
change record
1. record containing details of which configuration items are affected and how they are affected by an authorized change*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.3

3.389
change request
1. requests to expand or reduce the project scope, modify policies, processes, plans, or procedures, modify costs or budgets, or revise schedules*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.390
changeover system
1. a temporary information processing system used to facilitate the transition from an operational system to its successor. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System*

3.391
channel
1. a 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.8

NOTE 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.

3.392
channel capacity
1. the 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

3.393
character
1. a letter, digit, or other symbol that is used to represent information. ***2.*** a member of a set of elements that is used for the representation, organization, or control of data. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.11

3.394
character set
1. a collection of characters used in an encoding to represent terminal symbols. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE The character set used is significant in the encoding of text and string meta-attributes for a CDIF transfer.

3.395
character type
1. a 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

3.396
characteristic entity
1. a meta-entity that provides additional attribution for another meta-object. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

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

3.397
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

3.398
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

3.399
checkout
1. testing conducted in the operational or support environment to ensure that a software product performs as required after installation. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.5

3.400
checkpoint
1. a point in a computer program at which program state, status, or results are checked or recorded. ***2.*** an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.20

3.401
checkpointing
1. creating a checkpoint. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.21

NOTE Checkpoints can only be created when the engineering object involved satisfies a pre-condition stated in a checkpointing policy.

3.402
chief programmer
1. the 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

3.403
chief programmer team
1. a 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

3.404
child box
1. a box in a child diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.27

3.405
child diagram
1. a decomposition diagram related to a specific box by exactly one child/parent relationship. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.28

3.406
child entity
1. the entity in a specific relationship whose instances can be related to zero or one instance of the other entity (parent entity). *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.22

NOTE [key style]

3.407
CI
1. configuration item

3.408
CIM
1. Computer Integrated Manufacturing. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.409
clabject
1. dual entity that is a class and an object at the same time. *ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies*.3.13

NOTE 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).

3.410
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) — Fourth Edition*

3.411
class
1. an abstraction of the knowledge and behavior of a set of similar things. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.23.
2. a static programming entity in an object-oriented program that contains a combination of functionality and data. *Syn:* type

NOTE Classes are used to represent the notion of "things whose knowledge or actions are relevant".

3.412
class hierarchy
1. an ordering of classes, in which a subclass is a specialization of its superclass

NOTE A class inherits attributes and relationships from its superclass and can define additional attributes and relationships of its own.

3.413
classification
1. a choice within a category. *IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies*.3.3.
2. the manner in which the assets are organized for ease of search and extraction within a reuse library. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes*.3.4

3.414
classification process
1. a series of activities, starting with the recognition of an anomaly through to its closure. *IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies*.3.4

NOTE The process is divided into four sequential steps interspersed with three administrative activities. The sequential steps are as follows: a) Step 1: Recognition, b) Step 2: Investigation, c) Step 3: Action, d) Step 4: Disposition. The three administrative activities applied to each sequential step are as follows: a) Recording, b) Classifying, c) Identifying impact.

3.415
class-level attribute
1. a mapping from the class itself to the instances of a value class. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.24

3.416
class-level operation
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.25

3.417
class-level responsibility
1. a responsibility that represents some aspect of the knowledge, behavior, or rules of the class as a whole. *IEEE Std 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 registered- VoterCount for the class as a whole

3.418
clear
1. to set a variable, register, or other storage location to zero, blank, or other null value

cf. initialize, reset

3.419
clear text file encoding
1. a 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

3.420
client
1. the code or process that invokes an operation on an object. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.2.3

3.421
client-side
1. a 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

3.422
cloning
1. instantiating a cluster from a cluster checkpoint. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture*.8.1.24

3.423
close procurements
1. [Process] the process of completing each project procurement. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.424
close project or phase
1. [Process] the process of finalizing all activities across all of the project management process groups to formally complete the project or phase. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.425
closed loop
1. a 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

3.426
closed subroutine
1. a 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

3.427
closed term
1. a 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. *Syn:* ground term

3.428
closing processes
1. [Process Group] those processes performed to finalize all activities across all project management process groups to formally close the project or phase. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.429
cluster
1. a configuration of basic engineering objects forming a single unit for the purposes of deactivation, checkpointing, reactivation, recovery and migration. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.2

EXAMPLE a segment of virtual memory containing objects

3.430
cluster checkpoint
1. a cluster template containing checkpoints of the basic engineering objects in a cluster. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.22

3.431
cluster manager
1. an engineering object which manages the basic engineering objects in a cluster. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.3

3.432
cluster template
1. an object template for a configuration of objects and any activity required to instantiate those objects and establish initial bindings. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.19

3.433
CM
1. configuration management

3.434
CMIP
1. Common Management Information Protocol. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.435
CMIR
1. Client Makes it Right. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.436
CMIS
1. Common Management Information Service. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.437
CNCS
1. Client Native Code Set. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.438
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.*** a character or bit pattern that is assigned a particular meaning

cf. source code, object code, machine code, micro code

EXAMPLE a status code

3.439
code breakpoint
1. a breakpoint that is initiated upon execution of a given computer instruction. *Syn:* control breakpoint

cf. data breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint

3.440
code freeze
1. a period during which non-critical changes to the code are not allowed

3.441
code generator
1. a 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

3.442
code of accounts
1. [Tool] any numbering system used to uniquely identify each component of the work breakdown structure*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. chart of accounts

3.443
code of ethics standard
1. a standard that describes the characteristics of a set of moral principles dealing with accepted standards of conduct by, within, and among professionals

3.444
code review
1. a 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

3.445
code tuning
1. the 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

3.446
coding
1. in software engineering, the process of expressing a computer program in a programming language. ***2.*** the transforming of logic and data from design specifications (design descriptions) into a programming language

3.447
coerce
1. to treat an object of one type as an object of another type by using a different object*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.28

cf. cast

3.448
cohesion
1. the 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*. Syn:* module strength

cf. coupling

NOTE Types include coincidental, communicational, functional, logical, procedural, sequential, and temporal.

3.449
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

3.450
collaboration
1. the cooperative exchange of requests among classes and instances in order to achieve some goal*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.29

3.451
collapse
1. to terminate development on one branch by integrating it with another

3.452
collect requirements
1. [Process] the process of defining and documenting stakeholders' needs to meet the project objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.453
collection cardinality
1. a specification, for a collection-valued property, of how many members the value of the property, that is, the collection, may or must have for each instance*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.30

cf. cardinality constraint

3.454
collection class
1. a class in which each instance is a group of instances of other classes*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.31

3.455
collection-valued
1. a value that is complex*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.33

cf. scalar

NOTE That is, having constituent parts

3.456
collection-valued class
1. a class in which each instance is a collection of values*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.34

cf. scalar-valued class

3.457
collection-valued property
1. a property that maps to a collection class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.35. *Syn:* collection property

cf. scalar-valued property

3.458
co-location
1. [Technique] 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) — Fourth Edition*

3.459
command
1. an 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

3.460
command language
1. a language used to express commands to a computer system

cf. command-driven

3.461
command-driven
1. pertaining to a system or mode of operation in which the user directs the system through commands. *Syn:* command driven

cf. menu-driven

3.462
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

3.463
commercial-off-the-shelf (COTS)
1. software defined by a market-driven need, commercially available, and whose fitness for use has been demonstrated by a broad spectrum of commercial users. *IEEE Std 1062, 1998 Edition (R2002) IEEE Recommended Practice for Software Acquisition (includes IEEE Std 1062a).*3.3. *ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.4.3.* ***2.*** software product available for purchase and use without the need to conduct development activities. *ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.4; *ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.3. ***3.*** an item that a supplier offers to several acquirers for general use. *ISO/IEC 15289:2006, Systems and software engineering--Contents of systems and software life cycle information products (Documentation).* 5.2

cf. software product

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

3.464
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

3.465
commit message
1. an explanatory message accompanying a commit

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

3.466
commit privileges
1. a person's authority to commit changes

NOTE Sometimes privileges are associated with a specific part of the product (for example, artwork or documentation) or a specific branch.

3.467
commit war
1. a series of conflicting and mutually reversing commits introduced by developers who disagree on how a particular element should be coded

NOTE sometimes starts with a hostile backout

3.468
commit window
1. a period during which commits are allowed for a specific branch

NOTE In some development environments, commit windows for a maintenance branch might only open for short periods a few times a year.

3.469
commitment
1. an 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 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language*.6.5.2

NOTE The enterprise object(s) participating in an action of commitment may 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.

3.470
committer
1. a developer with commit privileges

3.471
common ancestor constraint
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.36

3.472
common cause
1. a source of variation that is inherent in the system and predictable. 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*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* random cause ***2.*** a source of variation of a process that exists because of normal and expected interactions among components of a process

cf. special cause.

3.473
common storage
1. a portion of main storage that can be accessed by two or more modules in a software system. *Syn:* common area, common block

cf. global data

3.474
common-environment coupling
1. a type of coupling in which two software modules access a common data area. *Syn:* common coupling, common environment coupling

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

3.475
communication interface
1. an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.14

3.476
communication management plan
1. [Output/Input] the document that describes: the communications needs and expectations for the project; how and in what format information will be communicated; when and where each communication will be made; and who is responsible for providing each type of communication. The communication management plan is contained in, or is a subsidiary plan of, the project management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.477
communicational cohesion
1. a 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

3.478
communications domain
1. a set of protocol objects capable of interworking. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.13

3.479
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

3.480
community
1. a configuration of objects formed to meet an objective. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*5.1.1

NOTE: The objective is expressed as a contract which specifies how the objective can be met.

3.481
community object
1. a composite enterprise object that represents a community. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.2.2

NOTE Components of a community object are objects of the community represented.

3.482
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

3.483
comparator
1. a software tool that compares two computer programs, files, or sets of data to identify commonalities or differences

NOTE Typical objects of comparison are similar versions of source code, object code, database files, or test results.

3.484
compatibility
1. the ability of two or more systems or components to perform their required functions while sharing the same hardware or software environment ***2.*** the ability of two or more systems or components to exchange information. ***3.*** the capability of a functional unit to meet the requirements of a specified interface without appreciable modification*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.11

3.485
competent
1. having the combination of knowledge, formal and informal skills, training, experience, and behavioral attributes required to perform a task or role*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.4

3.486
competent assessor
1. an assessor who has demonstrated the competencies to conduct an assessment and to monitor and verify the conformance of a process assessment*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.20

3.487
compile
1. to translate a computer program expressed in a high-order language into its machine language equivalent

cf. assemble, decompile, interpret

3.488
compile-and-go
1. an operating technique in which there are no stops between the compiling, linking, loading, and execution of a computer program

3.489
compiler
1. a computer program that translates programs expressed in a high-order language into their machine language equivalents

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

3.490
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

3.491
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

3.492
compiler generator
1. a translator or interpreter used to construct part or all of a compiler. *Syn:* compiler compiler, metacompiler

3.493
complete
1. includes necessary, relevant requirements or descriptive material, responses are defined for the range of valid input data, and terms and units of measure are defined. *IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*H.3 b)

3.494
complete ICOM code
1. a diagram feature reference in which dot notation joins an ICOM code to a diagram reference. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.29

3.495
complete procedure
1. all those activities which commence with entry to the procedure and conclude with exit from the procedure. *ISO 6593:1985 Information processing — Program flow for processing sequential files in terms of record groups.*2.1

3.496
complete table
1. a decision table where for all combinations of condition entries there exists a satisfying rule. *ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.17

NOTE In practical terms extended entry tables will include limited entries and are therefore mixed entry tables. Any extended or mixed entry table may be transformed into a limited entry table.

3.497
completion code
1. a code communicated to a job stream processor by a batch program to influence the execution of succeeding steps in the input stream

3.498
completion time theorem
1. a real-time scheduling theorem

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

3.499
complex processing GSC
1. one of the 14 general system characteristics describing the degree to which processing logic influences the development of the application. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.500
complexity
1. the 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.*** the degree to which a system or component has a design or implementation that is difficult to understand and verify

cf. simplicity

3.501
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 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.

3.502
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*

3.503
component
1. an entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis*. ISO/IEC 15026:1998, Information technology — System and software integrity levels*.3.1. ***2.*** one of the parts that make up a system*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.1.6. ***3.*** set of functional services in the software, which, when implemented, represents a well-defined set of functions and is distinguishable by a unique name*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method*.A.4

NOTE A component may be hardware or software and may be subdivided into other components. 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. A component may or may not be independently managed from the end-user or administrator's point of view.

3.504
component integration test
1. testing of groups of related components*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.1.7

3.505
component standard
1. a standard that describes the characteristics of data or program components subdivided into other components

3.506
component testing
1. testing of individual hardware or software components*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.1.8

3.507
composite key
1. a key comprising of two or more attributes*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.38

NOTE [key style]

3.508
composite task
1. a task containing nested objects

3.509
composite type
1. a 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)

3.510
computational interface template
1. an interface template for either a signal interface, a stream interface, or an operation interface*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.10

NOTE A computational interface template comprises a signal, a stream or an operation interface signature as appropriate, a behavior specification and an environment contract specification.

3.511
computational object template
1. an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.9

3.512
computational viewpoint
1. a 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.1.1.3

3.513
computer
1. a functional unit that can perform substantial computations, including numerous arithmetic operations and logic operations without human intervention*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.03

NOTE A computer may consist of a stand-alone unit or several interconnected units.

3.514
computer center
1. a facility that includes personnel, hardware, and software, organized to provide information processing services*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.19. *Syn:* data processing center

3.515
computer crime
1. a crime committed through the use, modification, or destruction of hardware, software, or data*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.07.02

3.516
computer generation
1. category in a historical classification of computers based mainly on the technology used in their manufacture*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.17

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

3.517
computer graphics
1. methods and techniques for construction, manipulation, storage, and display of images by means of a computer. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.08

3.518
computer instruction
1. a 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

3.519
computer language
1. a language designed to enable humans to communicate with computers

cf. design language, query language, programming language

3.520
computer network
1. a network of data processing nodes that are interconnected for the purpose of data communication. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.45

3.521
computer performance evaluation
1. an 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

3.522
computer program
1. a combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions ***2.*** a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.01

cf. software

3.523
computer program abstract
1. a 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

3.524
computer resource
1. an element of a data processing system needed to perform required operations. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.23

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

3.525
computer resource allocation
1. the 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

3.526
computer resources
1. the computer equipment, programs, documentation, services, facilities, supplies, and personnel available for a given purpose

cf. computer resource allocation

3.527
computer science
1. the branch of science and technology that is concerned with information processing by means of computers. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.18

3.528
computer software component (CSC)
1. a functionally or logically distinct part of a computer software configuration item, typically an aggregate of two or more software units

3.529
computer software configuration item (CSCI)
1. an aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process

cf. hardware configuration item, configuration item

3.530
computer system
1. a system containing one or more computers and associated software

3.531
computer-aided (CA)
1. pertaining to a technique or process in which a computer does part of the work. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.14

3.532
computer-aided design (CAD)
1. the 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 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.

3.533
computer-aided software engineering (CASE)
1. the use of computers to aid in the software engineering process. *Syn:* computer aided software engineering

NOTE may include the application of software tools to software design, requirements tracing, code production, testing, document generation, and other software engineering activities.

3.534
computer-based software system (CBSS)
1. a software system running on a computer*. ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.5

NOTE A CBSS may 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.

3.535
computerization
1. automation by means of computers*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.16

3.536
computerize
1. to automate by means of computers*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.15

3.537
computing center
1. a 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

3.538
computing system specification concepts
1. visible and quantifiable abstractions of computing system characteristics having attributes in isolation and relationships in context*. IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description.*3.4

3.539
computing system tool
1. a computer-based tool used by a developer or maintainer organization for creating and evolving dynamic systems*. IEEE Std 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

3.540
concept of operations (ConOps) document
1. a user-oriented document that describes a system's operational characteristics from the end user's viewpoint*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.4. *Syn:* operational concept description (OCD)

3.541
concept phase
1. the period of time in the system life cycle during which the user needs are identified and system concepts are described and evaluated

NOTE precedes the requirements phase

3.542
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*

3.543
conceptual model
1. a model of the concepts relevant to some endeavor*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.39

3.544
conceptual system design
1. a 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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.03.02

3.545
conciseness
1. software attributes that provide implementation of a function with a minimum amount of code

3.546
concurrency
1. the property of a system in which events may 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

3.547
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.*** a 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. *Syn:* parallel (2)

cf. simultaneous

NOTE A concurrent system or application has many threads of control.

3.548
concurrent communication diagram
1. a 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

3.549
concurrent enabling (of transition modes)
1. a 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

3.550
concurrent task architecture
1. a description of the concurrent tasks in a system or subsystem in terms of their interfaces and interconnections

3.551
condition
1. a 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

3.552
condition entry
1. an indication of the relevance of a condition to a particular rule*. ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.8

3.553
condition stub
1. a 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

3.554
conditional information
1. information supplied with every product to which it is relevant

3.555
conditional jump
1. a jump that takes place only when specified conditions are met

cf. unconditional jump

3.556
conduct procurements
1. [Process] the process of obtaining seller responses, selecting a seller, and awarding a contract*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.557
configuration
1. the 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.*** the arrangement of a system or network as defined by the nature, number, and chief characteristics of its functional units. ***4.*** the 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.26

cf. configuration item; form, fit, and function; version

3.558
configuration baseline
1. configuration information formally designated at a specific time during a product's or product component's life

NOTE Configuration baselines, plus approved changes from those baselines, constitute the current configuration information.

3.559
configuration control
1. an element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification. *Syn:* change control

cf. configuration identification, configuration status accounting

3.560
configuration control board (CCB)
1. a group of people responsible for evaluating and approving or disapproving proposed changes to configuration items, and for ensuring implementation of approved changes ***2.*** qualified personnel who evaluate, for approval or disapproval, all proposed changes to the current developmental baseline. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.08. *Syn:* change control board

cf. configuration control

3.561
configuration identification
1. an element of configuration management, consisting of selecting the configuration items for a system and recording their functional and physical characteristics in technical documentation ***2.*** the 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

3.562
configuration index
1. a 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

3.563
configuration item (CI)
1. 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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.7. ***2.*** item or aggregation of hardware or software or both that is designed to be managed as a single entity. *ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 2. ***3.*** component of an infrastructure or an item which is, or will be, under the control of configuration management. *ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.4. ***4.*** an aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process ***5.*** aggregation of work products that is designated for configuration management and treated as a single entity in the configuration management process

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

NOTE Configuration items may 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.

3.564
configuration item development record
1. a 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

3.565
configuration management (CM)
1. a discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements ***2.*** technical and organizational activities comprising configuration identification, control, status accounting, and auditing. *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*4.9

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

3.566
configuration management database (CMDB)
1. database containing all the relevant details of each configuration item and details of the important relationships between them*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.5

3.567
configuration management system
1. the 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.*** [Tool] 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) — Fourth Edition*

3.568
configuration status accounting
1. an element of configuration management, consisting of the recording and reporting of information needed to manage a configuration effectively

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

NOTE This information includes a listing of the approved configuration identification, the status of proposed changes to the configuration, and the implementation status of approved changes.

3.569
conflict
1. a change in one version of a file that cannot be reconciled with the version of the file to which it is applied

NOTE can occur when versions from different branches are merged or when two committers work concurrently on the same file

3.570
conformance
1. the fulfillment by a product, process or service of specified requirements*. IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*1.4.1

3.571
conformity evaluation report
1. document that describes the conduct and results of the evaluation carried out for a COTS software product*. ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.2

3.572
connectivity
1. the capability of a system or device to be attached to other systems or devices without modification*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.27

3.573
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.

3.574
consequence
1. an outcome of an event*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.1

NOTE The outcome may be a loss or a gain and may be expressed qualitatively or quantitatively.

3.575
consistency
1. the 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

3.576
consistent
1. Without internal conflicts*. IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*H.3 d)

3.577
constant
1. a quantity or data item whose value cannot change. ***2.*** an instance whose identity is known at the time of writing*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).* 3.1.40. ***3.*** the specification that an attribute or participant property value, once assigned, may not be changed, or that an operation shall always provide the same output argument values given the same input argument values*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).* 3.1.40. ***4.*** a numeric or string value that does not change during program execution

cf. variable figurative constant, literal

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

NOTE The identity of a constant state class instance is represented by #K, where K is an integer or a name.

3.578
constant-failure period
1. the 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

3.579
constraint
1. a limitation or implied requirement that constrains the design solution or implementation of the systems engineering process and is not changeable by the enterprise*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.5. ***2.*** a restriction on software life cycle process (SLCP) development*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*Annex E. ***3.*** a rule that specifies a valid condition of data*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.41. ***4.*** a responsibility that is a statement of facts that are required to be true in order for the constraint to be met*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.41. ***5.*** a 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.*** an externally imposed limitation on system requirements, design, or implementation or on the process used to develop or modify a system*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.5. ***7.*** [Input]. the state, quality, or sense of being restricted to a given course of action or inaction. An applicable restriction or limitation, either internal or external to a project, which will affect the performance of the project or a process. For example, a schedule constraint is any limitation or restraint placed on the project schedule that affects when a schedule activity can be scheduled and is usually in the form of fixed imposed dates*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***8.*** a statement that expresses measurable bounds for an element or function of the system*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.4

cf. software life cycle process (SLCP)

NOTE That is, a constraint is a factor that is imposed on the solution by force or compulsion and may limit or modify the design changes.

3.580
construction
1. the process of writing, assembling, or generating assets*. IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology - Software Life Cycle Processes - Reuse Processes.*3.5. ***2.*** the activity in software development consisting of detailed design, coding, unit testing, and debugging

NOTE the collection of activities focused on creating source code

3.581
consumer
1. the organization or person who buys the software package*. ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.2

3.582
consumer software package
1. a software product designed and sold to carry out identified functions; the software and its associated documentation are packaged for sale as a unit*. ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.1

3.583
container interface
1. an interface of a data repository allowing access to data*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*14.1.1.2

3.584
content coupling
1. a 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

3.585
context
1. the immediate environment in which a function (or set of functions in a diagram) operates*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0.*2.1.30

3.586
context diagram
1. a 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0.*2.1.31

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

3.587
context of use
1. users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*.4.7

NOTE [ISO 9241-11:1998]

3.588
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.10

cf. embedded documentation, printed documentation

3.589
contiguous allocation
1. a 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)

3.590
contingency plan
1. a plan for dealing with a risk factor, should it become a problem

3.591
contingency reserve
1. [Output/Input] the amount of funds, budget, or time needed above the estimate to reduce the risk of overruns of project objectives to a level acceptable to the organization*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.592
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

3.593
continuous iteration
1. a loop that has no exit

3.594
continuous representation
1. capability maturity model structure wherein capability levels provide a recommended order for approaching process improvement within each specified process area

3.595
continuous risk management
1. the process of analyzing the progress of a planned activity, project, or program on a periodic, ongoing basis and handling identified risk factors

NOTE Includes developing options and fallback positions to permit alternative solutions to reduce the impact if a risk factor becomes a problem.

3.596
contract
1. binding agreement between two parties, especially enforceable by law, or a similar internal agreement wholly within an organization. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.8. ***2.*** a legally binding document agreed upon by the customer and supplier. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.*3.1. ***3.*** [Output/Input] a contract is 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) — Fourth Edition*

3.597
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

3.598
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

3.599
contract work breakdown structure (CWBS)
1. portion of the overall work breakdown structure applicable to a contract, developed and maintained by the supplier

3.600
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

3.601
contravariance
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.42

3.602
contribution
1. the function type's (ILF, EIF, EI, EO, EQ) contribution to the unadjusted function point count. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.603
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) — Fourth Edition.* ***3.*** in an IDEF0 model, a condition or set of conditions required for a function to produce correct output. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*2.1.32

3.604
control account
1. [Tool]. a management control point where scope, budget (resource plans), 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) — Fourth Edition*

cf. work package

3.605
control arrow
1. an arrow or arrow segment that expresses IDEF0 control*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.33

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

3.606
control chart
1. [Tool] a graphic display of process data over time and against established control limits, and that 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) — Fourth Edition*

3.607
control clustering
1. a task-structuring criterion by which a control object is combined into a task with the objects it controls

3.608
control costs
1. [Process] the process of monitoring the status of the project to update the project budget and managing changes to the cost baseline*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.609
control coupling
1. a 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

3.610
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

3.611
control field
1. the field comprising one or more input variables whose change in value, or lack of change, between successive logical records affect the flow of control through the main procedure*. ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.6

3.612
control flow
1. the sequence in which operations are performed during the execution of a computer program. *Syn:* flow of control

cf. data flow

3.613
control flow diagram
1. a diagram that depicts the set of all possible sequences in which operations may be performed during the execution of a system or program

cf. data flow diagram, call graph, structure chart

NOTE Types include box diagram, flowchart, input-process-output chart, state diagram.

3.614
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 of the application being counted*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6

NOTE It specifies what, when, or how data is to be processed.

3.615
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 that reflects the expected variation in the data*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. specification limits

3.616
control loopback
1. loopback of output from one function to be control for another function in the same diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.34. *Syn:* feedback

3.617
control point
1. a project agreed on point in time or times when specified agreements or controls are applied to the software configuration items being developed*. IEEE Std 828-2005 IEEE Standard for Software Configuration Management Plans.*2.1.1. *Syn:* project control point

EXAMPLE an approved baseline or release of a specified document/code or project milestone

3.618
control schedule
1. [Process] the process of monitoring the status of the project to update project progress and managing changes to the schedule baseline*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.619
control scope
1. [Process] 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) — Fourth Edition*

3.620
control statement
1. a 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

3.621
control store
1. in a microprogrammed computer, the computer memory in which microprograms reside

cf. microword, nanostore

3.622
control task
1. a task that makes decisions to control other tasks' execution

3.623
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

3.624
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

3.625
conversion
1. modification of existing software to enable it to operate with similar functional capability in a different environment. ***2.*** those activities associated with mapping data or programs from one format to another. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

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

3.626
conversion functionality
1. for a development project, functions provided to convert data and/or provide other user-specified conversion requirements, such as special conversion reports. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE for an enhancement project, functions delivered because of any conversion functionality required by the user

3.627
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

3.628
cookie
1. a quantity used to indicate or signal to a recipient of data, significant changes in the state of the entity supplying the data. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.3

NOTE Web sites may store/retrieve cookies from user client systems to maintain state information including identification of users and transaction coherency.

3.629
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 may differ from that of the source ***2.*** the result of a copy process as in (1)

cf. move

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

3.630
copyright
1. the 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

3.631
COQ
1. cost of quality. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.632
CORBA
1. Common Object Request Broker Architecture. *ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*4

3.633
coroutine
1. a 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

3.634
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 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 3

3.635
corporate executive level
1. the management level responsible for the enterprise, including the Board of Directors. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.636
corporate governance
1. system by which organizations are directed and controlled. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.2

3.637
corporate governance of IT
1. system by which the current and future use of IT is directed and controlled. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.3

NOTE Corporate governance of IT involves evaluating and directing the use of IT to support the organization and monitoring this use to achieve plans. It includes the strategy and policies for using IT within an organization.

3.638
correctability
1. the degree of effort required to correct software defects and to cope with user complaints

3.639
corrective action
1. documented direction for executing the project work to bring expected future performance of the project work in line with the project management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** acts used to remedy a situation, remove an error, or adjust a condition

3.640
corrective maintenance
1. the reactive modification of a software product performed after delivery to correct discovered problems. *ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.2 ***2.*** maintenance performed to correct faults in hardware or software

NOTE The modification repairs the software product to satisfy requirements.

3.641
correctness
1. the degree to which a system or component is free from faults in its specification, design, and implementation. ***2.*** the degree to which software, documentation, or other items meet specified requirements. ***3.*** the degree to which software, documentation, or other items meet user needs and expectations, whether specified or not

3.642
COSMIC
1. Common Software Measurement International Consortium. *ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method*

3.643
cost constraint
1. any limitation or restraint placed on the project budget such as funds available over time. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.644
cost management plan
1. [Output/Input] the document that sets out the format and establishes the activities and criteria for planning, structuring, and controlling the project costs. The cost management plan is contained in, or is a subsidiary plan, of the project management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.645
cost of quality (COQ)
1. [Technique] 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) — Fourth Edition*

3.646
cost performance baseline
1. a specific version of the time-phased budget used to compare actual expenditures to planned expenditures to determine if preventive or corrective action is needed to meet the project objectives. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.647
cost performance index (CPI)
1. a measure of cost efficiency on a project. It is the ratio of earned value (EV) to actual costs (AC). CPI = EV divided by AC. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.648
cost variance (CV)
1. a measure of cost performance on a project. It is the difference between earned value (EV) and actual cost (AC). CV = EV minus AC*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.649
cost-plus-fee (CPF)
1. a contract in which the acquirer reimburses the supplier's allowable costs for performing the contract work and also pays a fee. *Syn:* cost plus fee

3.650
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) — Fourth Edition*

3.651
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) — Fourth Edition*

3.652
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) — Fourth Edition*

3.653
COTS
1. Commercial-Off-The-Shelf*. ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.4

3.654
counter
1. a 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

3.655
counting rule
1. conditions and procedures under which the measurement value is obtained*. ISO/IEC 14598-3:2000, Software engineering — Product evaluation — Part 3: Process for developers.*4.1

3.656
counting scope
1. the counting scope defines the functionality which will be included in a particular function point count*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.657
coupling
1. the manner and degree of interdependence between software modules. ***2.*** the strength of the relationships between modules. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)*.1.4.2. ***3.*** a 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

cf. cohesion

NOTE Types include common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling, and pathological coupling.

3.658
CPAF
1. cost plus award fee. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.659
CPC
1. computer program component

cf. computer software component

3.660
CPCI
1. computer program configuration item

cf. computer software configuration

3.661
CPF
1. cost-plus-fee. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.662
CPFF
1. cost plus fixed fee. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.663
CPI
1. cost performance index. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.664
CPIF
1. cost plus incentive fee. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.665
CPM
1. critical path methodology. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***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*

NOTE the IFPUG equivalent of the NESMA International Standard ISO/IEC 24570:2005, Definitions and Counting Guidelines for the Application of Function Point Analysis

3.666
CPPC
1. cost plus percentage of cost

3.667
crash
1. the sudden and complete failure of a computer system or component

cf. hard failure

3.668
crashing
1. [Technique] a specific type of project schedule compression technique performed by taking action to decrease the total project schedule duration after analyzing a number of alternatives to determine how to get the maximum schedule duration compression for the least additional cost. Typical approaches for crashing a schedule include reducing schedule activity durations and increasing the assignment of resources on schedule activities*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. fast tracking, schedule compression

3.669
create WBS (work breakdown structure)
1. [Process] 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) — Fourth Edition*

3.670
crisis
1. a critical state of affairs in which a decisive, probably undesirable outcome is impending

3.671
crisis management
1. steps to take when a contingency plan does not solve the associated problem

3.672
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) — Fourth Edition.* ***2.*** specific data items identified as contents of information items for appraising a factor in an evaluation, audit, test or review. *ISO/IEC 15289:2006, Systems and software engineering — Contents of systems and software life cycle information products*. 5.2

3.673
critical activity
1. any schedule activity on a critical path in a project schedule. Most commonly determined by using the critical path method. Although some activities are "critical ", in the dictionary sense, without being on the critical path, this meaning is seldom used in the project context*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.674
critical chain method
1. [Technique] a schedule network analysis technique that modifies the project schedule to account for limited resources*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.675
critical design review (CDR)
1. a 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.*** a review as in (1) of any hardware or software component

3.676
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; Systems and software engineering — Requirements for designers and developers of user documentation.*4.11

3.677
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

3.678
critical path
1. generally, but not always, the sequence of schedule activities that determines the duration of the project. It is the longest path through the project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. critical path methodology

3.679
critical path methodology (CPM)
1. [Technique] a schedule network analysis technique used to determine the amount of scheduling flexibility (the amount of float) on various logical network paths in the project schedule network, and to determine the minimum total project duration. Early start and finish dates are calculated by means of a forward pass, using a specified start date. Late start and finish dates are calculated by means of a backward pass, starting from a specified completion date, which sometimes is the project early finish date determined during the forward pass calculation*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. critical path

3.680
critical piece first
1. a system development approach in which the most critical aspects of a system are implemented first

cf. bottom-up, top-down

NOTE The critical piece may be defined in terms of services provided, degree of risk, difficulty, or other criteria.

3.681
critical range
1. metric values used to classify software into the categories of acceptable, marginal, or unacceptable*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.2

3.682
critical section
1. the section of a task's internal logic that is executed mutually exclusively with other tasks

3.683
critical value
1. metric value of a validated metric that is used to identify software that has unacceptable quality*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.3

3.684
criticality
1. the degree of impact that a requirement, module, error, fault, failure, or other characteristic has on the development or operation of a system*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.9

3.685
cross-assembler
1. an assembler that executes on one computer but generates machine code for a different computer

3.686
cross-compiler
1. a compiler that executes on one computer but generates machine code for a different computer

3.687
cross-reference generator
1. a 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. *Syn:* cross-referencer

3.688
cross-reference list
1. a 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

3.689
cross-reference tool
1. a software maintenance tool that lets the user determine where a variable is used or where a particular procedure is called on

3.690
CSC
1. computer software component

3.691
CSCI
1. computer software configuration item

3.692
CSF
1. critical success factor*. ISO/IEC TR 14471:2007, Information technology — Software engineering — Guidelines for the adoption of CASE tools.*2.2

3.693
CT
1. communication technology*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.7. *Syn:* Communication Technology

3.694
curriculum standard
1. a standard that describes the characteristics of a course of study on a body of knowledge that is offered by an educational institution

3.695
custom software
1. software product developed for a specific application from a user requirements specification*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.8

3.696
customer
1. organization or person that receives a product or service*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes*.4.9; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes*.4.9. ***2.*** the entity or entities for whom the requirements are to be satisfied in the system being defined and developed*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications*.3.5. ***3.*** an individual or organization who acts for the ultimate user of a new or modified hardware or software product to acquire the product and its documentation*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology-System Definition -Concept of Operation Document*.3.7. ***4.*** the person, or persons, who pay for the product and usually (but not necessarily) decide the requirements*. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications*.3.2. *Syn:* acquirer, buyer, beneficiary, purchaser

cf. stakeholder

EXAMPLE an end-user of the completed system, an organization within the same company as the developing organization (e.g., System Management)

NOTE A customer can be internal or external to the organization. The customer may be a higher level project. 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.

3.697
customer requirement
1. the 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

3.698
customization
1. adaptation of a software or documentation product to the needs of a particular audience*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation*.4.12

3.699
cut-off date
1. date after which changes to the software are reflected in the

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

3.700
CWBS
1. contract work breakdown structure

3.701
cycle
1. a period of time during which a set of events is completed. ***2.*** a set of operations that is repeated regularly in the same sequence, possibly with variations in each repetition

cf. software development cycle, software life cycle

3.702
cycle stealing
1. the 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

3.703
cyclic search
1. a storage allocation technique in which each search for a suitable block of storage begins with the block following the one last allocated

3.704
data
1. a 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 15939:2007, Systems and software engineering — Measurement process.*3.4. ***2.*** collection of values assigned to base measures, derived measures and/or indicators. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.9. ***3.*** the representation forms of information dealt with by information systems and users thereof. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.6. ***4.*** a reinterpretable representation of information in a formalized manner suitable for communication, interpretation, or communication, or processing. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.02

cf. data type

3.705
data abstraction
1. the process of extracting the essential characteristics of data by defining data types and their associated functional characteristics and disregarding representation details ***2.*** the result of the process in (1)

cf. encapsulation, information hiding

3.706
data analysis
1. a systematic investigation of the data and their flow in a real or planned system. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.08

3.707
data attribute
1. the smallest parcel of information, within an identified data group, carrying a meaning from the perspective of the software's functional user requirements. *ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.4. ***2.*** a characteristic of an entity. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.708
data bank
1. a set of data related to a given subject and organized in such a way that it can be consulted by subscribers. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.08.04

3.709
data breakpoint
1. a breakpoint that is initiated when a specified data item is accessed. *Syn:* storage breakpoint

cf. code breakpoint, dynamic breakpoint, epilog breakpoint, programmable breakpoint, prolog breakpoint, static breakpoint

3.710
data characteristic
1. an inherent, possibly accidental, trait, quality, or property of data. *IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

EXAMPLE arrival rates, formats, value ranges, or relationships between field values

3.711
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.39

3.712
data communications GSC
1. one of the 14 general system characteristics describing the degree to which the application communicates directly with the processor. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.713
data coupling
1. a type of coupling in which output from one software module serves as input to another module. *Syn:* input-output coupling

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

3.714
data date
1. the date up to or through which the project's reporting system has provided actual status and accomplishments. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* as-of date, time-now date

3.715
data declaration source statement
1. source statement that reserves or initializes memory at compilation time

3.716
data element
1. smallest parcel of information, within an identified data group, carrying a meaning from the perspective of the software's Functional User Requirements (FUR). *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.7. *Syn:* data item

3.717
data element type (DET)
1. a unique, user-recognizable, non-recursive item of information. *ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10. ***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-repeated field. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6. ***4.*** unique, user recognizable, non-repeated field in a BFC. *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.3

NOTE A data element can be a character string, or a digital or graphical element in a BFC. When 'data elements' are indicated for a BFC, the number of data elements is always greater than 0.

3.718
data exception
1. an exception that occurs when a program attempts to use or access data incorrectly

cf. addressing exception, operation exception, overflow exception, protection exception, underflow exception

3.719
data flow
1. the sequence in which data transfer, use, and transformation are performed during the execution of a computer program

cf. control flow

3.720
data flow diagram (DFD)
1. a 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. *Syn:* data flowchart, data flow graph

cf. control flow diagram, data structure diagram

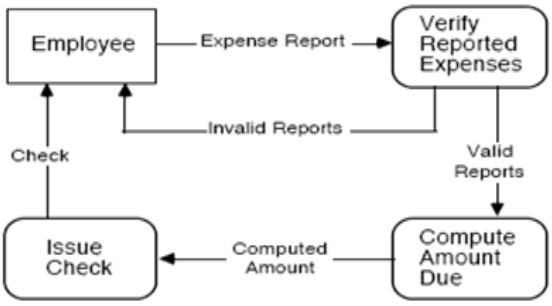


Figure 8 — Data flow diagram

3.721
data function
1. 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*

cf. data functions

NOTE 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.

3.722
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*

3.723
data functions
1. the functionality provided to the user to meet internal and external data requirements*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. data function

NOTE Data functions are either internal logical files (ILFs) or external interface files (EIFs).

3.724
data group
1. 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*

cf. object of interest

3.725
data group
1. 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 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.6

cf. object of interest

3.726
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*

3.727
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

3.728
data inventory
1. in an information processing system, all the data and their characteristics, including interdependencies. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.06

3.729
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.08.02. ***2.*** the disciplined processes and systems that plan for, acquire, and provide stewardship for business and technical data, consistent with data requirements, throughout the data lifecycle

3.730
data medium
1. a material in or on which data can be recorded and from which data can be retrieved. Plural: media. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.51

3.731
data model
1. a 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 13201998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.44. ***2.*** a 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

NOTE A data model is one that may 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]

3.732
data movement (-type)
1. a base functional component which moves one or more data attributes belonging to a single data group. *ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.6

NOTE There are four types of COSMIC-FFP data movements: Entry, Exit, Read and Write.

3.733
data processing (DP)
1. the systematic performance of operations upon data*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.06. *Syn:* automatic data processing (ADP)

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 The term data processing should not be used as a synonym for information processing.

3.734
data processing system
1. one or more computers, peripheral equipment, and software that perform data processing*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.20. *Syn:* computer system, computing system

3.735
data protection
1. the implementation of appropriate administrative, technical, or physical means to guard against unauthorized intentional or accidental disclosure, modification, or destruction of data*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.07.01

3.736
data provider
1. individual or organization that is a source of data*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.5

3.737
data repository
1. an object providing the storage function*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*14.1.1.1

3.738
data store
1. organized and persistent collection of data and information that allows for its retrieval*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.6

3.739
data structure
1. a physical or logical relationship among data elements, designed to support specific data manipulation functions

3.740
data structure diagram
1. a diagram that depicts a set of data elements, their attributes, and the logical relationships among them

cf. data flow diagram, entity-relationship diagram

<table>
<tr><td colspan="10">Employee Record</td></tr>
<tr><td rowspan="2">Emp. No. (4I)</td><td colspan="3">Emp. Name</td><td colspan="4">Emp. Address</td><td rowspan="2">Dept. No. (3I)</td><td rowspan="2">Emp. sal. (4I)</td></tr>
<tr><td>First (10C)</td><td>Mid. (1C)</td><td>Last (16C)</td><td>Street (20C)</td><td>City (20C)</td><td>State (2C)</td><td>Zip (9I)</td></tr>
</table>

Figure 9 — Data structure diagram

3.741
data structure-centered design
1. a 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

3.742
data type
1. a class of data, characterized by the members of the class and the operations that can be applied to them. ***2.*** a categorization of an abstract set of possible values, characteristics, and set of operations for an attribute. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.45. ***3.*** a set of values and operations on those values. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.45. ***4.*** a categorization of values operation arguments, typically covering both behavior and representation. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.4

EXAMPLE integers, real numbers, and character strings

NOTE [key style]

3.743
database
1. a collection of interrelated data stored together in one or more computerized files. ***2.*** a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.08.05. ***3.*** collection of data describing a specific target area that is used and updated by one or more applications. *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.5. *Syn:* data base

3.744
database design specification
1. a document that describes the content and format of the permanent or semi-permanent data necessary for the software to carry out its functions

3.745
data-sensitive fault
1. a fault that causes a failure in response to some particular pattern of data. *Syn:* pattern-sensitive fault

cf. program-sensitive fault

3.746
data-structure-oriented design
1. a design methodology used for business applications by basing the design on the logical data structures of the program specification

EXAMPLE the Jackson System Design and Warnier-Orr methods

3.747
datum
1. singular of ""data"

NOTE Data"" may be used for both singular and plural.

3.748
DCE
1. Distributed Computing Environment. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.749
deactivation
1. checkpointing a cluster, followed by deletion of the cluster. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.23

3.750
deadlock
1. a situation in which computer processing is suspended because two or more devices or processes are each awaiting resources assigned to the others ***2.*** a situation in which two or more tasks are suspended indefinitely because each task is waiting for a resource acquired by another task

cf. lockout

3.751
deblock
1. to separate the parts of a block

cf. block (2)

3.752
debug
1. to detect, locate, and correct faults in a computer program. ***2.*** to detect, locate, and eliminate errors in programs. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.07

NOTE Techniques include use of breakpoints, desk checking, dumps, inspection, reversible execution, single-step operation, and traces.

3.753
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 15939:2007, Systems and software engineering — Measurement process.*3.7

3.754
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 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.10

3.755
decision table
1. a 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.*** a table used to show sets of conditions and the actions resulting from them. ***3.*** a 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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.05

3.756
decision tree analysis
1. [Technique] the decision tree is a diagram that describes a decision under consideration and the implications of choosing one or another of the available alternatives. It is used when some future scenarios or outcomes of actions are uncertain. It incorporates probabilities and the costs or rewards of each logical path of events and future decisions, and uses expected monetary value analysis to help the organization identify the relative values of alternate actions*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.757
declaration
1. an action that establishes a state of affairs in the environment of the object making the declaration*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language*.6.5.3. ***2.*** a non-executable program statement that affects the assembler or compiler's interpretation of other statements in the program ***3.*** a 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 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.

3.758
declarative language
1. a 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

3.759
decompile
1. to translate a compiled computer program from its machine language version into a form that resembles, but may not be identical to, the original high-order language program

cf. compile

3.760
decompiler
1. a software tool that decompiles computer programs

3.761
decomposition
1. [Technique] a planning technique that subdivides the project scope and project deliverables into smaller, more manageable components, until the project work associated with accomplishing the project scope and providing the deliverables is defined in sufficient detail to support executing, monitoring, and controlling the work*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** the partitioning of a modeled function into its component functions*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.35

3.762
decomposition diagram
1. a diagram that details its parent box*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0.*2.1.36

3.763
decoupling
1. the 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

3.764
defect
1. a problem which, if not corrected, could cause an application to either fail or to produce incorrect results. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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) — Fourth Edition.* ***3.*** a generic term that can refer to either a fault (cause) or a failure (effect). *IEEE Std 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

3.765
defect density
1. number of defects per unit of product size

EXAMPLE problem reports per thousand lines of code

3.766
defect repair
1. formally documented identification of a defect in a project component with a recommendation to either repair the defect or completely replace the component. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.767
defensive programming
1. a general approach to programming that assumes that errors will occur during both initial development and maintenance and, as a result, creates code in such a way that the program still operates properly when errors occur

3.768
define activities
1. [Process] 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) — Fourth Edition*

3.769
define scope
1. [Process] the process of developing a detailed description of the project and product. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.770
defined process
1. a process that is managed (planned, monitored and adjusted), and tailored from the organization's set of standard processes according to the organization's tailoring guidelines. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.21

NOTE A defined process has a maintained process description; 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 project's tasks and activities of the project.

3.771
definitive master version
1. version of the software that is used to install the software and to create distribution copies. *ISO/IEC 19770-1:2006 Information technology — Software asset management — Part 1: Processes.*3. 4

3.772
degree of confidence
1. the degree of confidence that software conforms to its requirements. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.2

3.773
degree of influence (DI)
1. a numerical indicator of the amount of impact of each of the 14 general system characteristics, ranging from zero to five. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** a numerical indicator of the impact of each of the 19 (or more) technical complexity adjustment factors, ranging from 0 (no influence) to 5 (strong influence, throughout). *ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10

NOTE These indicators are used to compute the value adjustment factor.

3.774
delegation
1. the action that assigns authority, responsibility or a function to another object. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.5.4

NOTE A delegation, once made, may later be withdrawn.

3.775
deleted source statement
1. source statement that is removed or modified from an existing software product as a new product is constructed

3.776
delimiter
1. a character or set of characters used to denote the beginning or end of a group of related bits, characters, words, or statements

3.777
deliver primitive
1. a 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

3.778
deliverable
1. [Output/Input] any unique and verifiable product, result, or capability to perform a service that must be produced to complete a process, phase, or project. Often used more narrowly in reference to an external deliverable, which is a deliverable that is subject to approval by the project sponsor or customer. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** item to be provided to an acquirer or other designated recipient as specified in an agreement

cf. acquirer, product, result

NOTE This item can be a document, hardware item, software item, service, or any type of work product.

3.779
deliverables
1. items whose delivery to the customer is a requirement of the contract*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.9

3.780
delivered source statement
1. source statement that is incorporated into the product delivered to the customer

3.781
delivery
1. release of a system or component to its customer or intended user

cf. software life cycle, system life cycle

3.782
delivery rate
1. the productivity measure for creating or enhancing an application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE It is expressed by the Project Function Points divided by the Work Effort for the development or enhancement project.

3.783
Delphi technique
1. [Technique] 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) — Fourth Edition*

3.784
demand paging
1. a storage allocation technique in which pages are transferred from auxiliary storage to main storage only when those pages are needed

cf. anticipatory paging

3.785
demodularization
1. in software design, the process of combining related software modules, usually to optimize system performance

cf. downward compression, lateral compression, upward compression

3.786
demonstration
1. a 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

3.787
demonstrative product
1. a 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)

3.788
dependability
1. trustworthiness of a computer system such that reliance can be justifiably placed on the service it delivers. *IEEE Std 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability.*2.2
2. measure of the degree to which an item is operable and capable of performing its required function at any (random) time during a specified mission profile, given item availability at the start of the mission

NOTE Reliability, availability, and maintainability are aspects of dependability.

3.789
dependent entity
1. an entity for which the unique identification of an instance depends upon its relationship to another entity. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.46. *Syn:* identifier-dependent entity

cf. independent entity [key style]

NOTE Expressed in terms of the foreign key, an entity is said to be dependent if any foreign key is wholly contained in its primary key

3.790
dependent state class
1. a class whose instances are, by their very nature, intrinsically related to certain other state class instance(s). *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.47

cf. independent state class

NOTE It would not be appropriate to have a dependent state class instance by itself and unrelated to an instance of another class(es) and, furthermore, it makes no sense to change the instance(s) to which it relates.

3.791
deployment
1. phase of a project in which a system is put into operation and cutover issues are resolved

cf. release

3.792
derived data
1. data that requires processing other than or in addition to direct retrieval and validation of information from internal logical files and/or external interface files. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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*

3.793
derived measure
1. measure that is defined as a function of two or more values of base measures. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.8; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.11

NOTE A transformation of a base measure using a mathematical function can also be considered as a derived measure.

3.794
derived property
1. the designation given to a property whose value is determined by computation. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.50. *Syn:* derived attribute, derived participant party

NOTE The typical case of a derived property is as a derived attribute although there is nothing to prohibit other kinds of derived property.

3.795
derived requirement
1. a lower-level requirement that is determined to be necessary for a top-level requirement to be met. ***2.*** a 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 Derived requirements can arise during analysis and design of components of the product or service.

3.796
derived type
1. a data type whose members and operations are taken from those of another data type according to some specified rule

cf. subtype

3.797
descendent box
1. a box in a descendent diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.37

3.798
descendent diagram
1. a decomposition diagram related to a specific box by a hierarchically consecutive sequence of one or more child/parent relationships. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.38

3.799
description standard
1. a standard that describes the characteristics of product information or procedures provided to help understand, test, install, operate, or maintain the product

3.800
design
1. the process of defining the architecture, components, interfaces, and other characteristics of a system or component. ***2.*** the result of the process in (1). ***3.*** the process of defining the software architecture, components, modules, interfaces, and data for a software system to satisfy specified requirements ***4.*** the process of conceiving, inventing, or contriving a scheme for turning a computer program specification into an operational program ***5.*** activity that links requirements analysis to coding and debugging. ***6.*** stage of documentation development that is concerned with determining what documentation will be provided in a product and what the nature of the documentation will be. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.13

cf. architectural design, detailed design, preliminary design

3.801
design analyzer
1. an 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

3.802
design architecture
1. an arrangement of design elements that provides the design solution for a product or life cycle process intended to satisfy the functional architecture and the requirements baseline*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.7

3.803
design authority
1. the person or organization that is responsible for producing the design of the system*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.3

3.804
design characteristic
1. the design attributes or distinguishing features that pertain to a measurable description of a product or process*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.8

3.805
design concept
1. a fundamental idea that can be applied to designing a system

EXAMPLE information hiding

3.806
design constraint
1. a software requirement that impacts or constrains the design of a software system or software system component. *Syn:* software design requirement

EXAMPLE physical requirements, performance requirements, software development standards, and software quality assurance (SQA) standards

3.807
design description
1. a document that describes the design of a system or component. *Syn:* design document, design specification

cf. product specification, requirements specification

NOTE Typical contents include system or component architecture, control logic, data structures, input/output formats, interface descriptions, and algorithm.

3.808
design element
1. a basic component or building block in a design

3.809
design entity
1. part of a design that is structurally, functionally, or otherwise distinct from other elements or that plays a different role relative to other design entities ***2.*** an element (component) of a design that is structurally and functionally distinct from other elements and that is separately named and referenced*. IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions.*3.1

3.810
design fault
1. a design (specification, coding) fault that results from a human error during system design and that might result in a design failure

3.811
design language
1. a specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a hardware or software design ***2.*** a 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 Types include hardware design language, program design language.

3.812
design level
1. the design decomposition of the software item

EXAMPLE system, subsystem, program, or module

3.813
design methodology
1. a systematic approach to creating a design consisting of the ordered application of a specific collection of tools, techniques, and guidelines

3.814
design pattern
1. a description of the problem and the essence of its solution to enable the solution to be reused in different settings

NOTE not a detailed specification, but a description of accumulated wisdom and experience

3.815
design phase
1. the period in the software life cycle during which definitions for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements ***2.*** the period in the software life cycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements

cf. detailed design, preliminary design

3.816
design requirement
1. a requirement that specifies or constrains the design of a system or system component

cf. functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement

3.817
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.*** a 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 Types include critical design review, preliminary design review, system design review.

3.818
design standard
1. a standard that describes the characteristics of a design or a design description of data or program components

3.819
design strategy
1. the overall plan and direction for performing design

EXAMPLE functional decomposition

3.820
design unit
1. a logically related collection of design elements

NOTE In an Ada PDL, a design unit is represented by an Ada compilation unit.

3.821
design view
1. a subset of design entity attribute information that is specifically suited to the needs of a software project activity. *IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions*.3.2

3.822
design-to cost
1. cost goals for the production cost (design-to unit production cost) and life-cycle cost (design-to life-cycle cost) of the products used to make the design converge on cost targets. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process*.3.1.9. *Syn:* design to cost

cf. design-to-cost

NOTE Cost goals should be treated with the same attention as any other performance parameter.

3.823
design-to-cost
1. an approach to managing a system/software project so as to hold the project to a predetermined cost. *Syn:* cost as an independent variable (CAIV), design to cost

cf. design-to cost

NOTE Actual and projected costs are closely tracked, and actions such as deleting or postponing lower-priority requirements are taken if costs threaten to exceed targets.

3.824
desk checking
1. the 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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.02. ***2.*** a 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

3.825
desktop publishing
1. electronic publishing using a microcomputer. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.16

3.826
destination address
1. the address of the device or storage location to which data is to be transferred

cf. source address

3.827
destructive read
1. a read operation that alters the data in the accessed location

cf. nondestructive read

3.828
detailed design
1. the 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.*** the result of the process in (1)

cf. software development process

3.829
detailed design description
1. a document that describes the exact detailed configuration of a computer program. *Syn:* detailed design specification

NOTE It identifies the input, output, control logic, algorithms, and data structure of each individual low-level component of the software product and is the primary product of the detailed design phase.

3.830
detailed design phase
1. the 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

3.831
detailed design review
1. a 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

3.832
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 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.

3.833
determine budget
1. [Process] 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) — Fourth Edition*

3.834
develop human resource plan
1. [Process] the process of identifying and documenting project roles, responsibilities, and required skills, reporting relationships, and creating a staffing management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.835
develop project charter
1. [Process] the process of developing a document that formally authorizes a project or a phase and documenting initial requirements that satisfy the stakeholders' needs and expectations. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.836
develop project management plan
1. [Process] the process of documenting the actions necessary to define, prepare, integrate, and coordinate all subsidiary plans*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.837
develop project team
1. [Process] 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) — Fourth Edition*

3.838
develop schedule
1. [Process] the process of analyzing activity sequences, durations, resource requirements, and schedule constraints to create the project schedule*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.839
developed source statement
1. source statement that is newly created for, added to, or modified for a software product

3.840
developer
1. organization that performs development tasks (including requirements analysis, design, testing through acceptance) during a life cycle process*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.10. ***2.*** person who applies a methodology for some specific job, usually an endeavor*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.11

NOTE May include 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. Developers apply methodologies via enactment.

3.841
development
1. the 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.*** the specification, construction, testing, and delivery of a new information system. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***3.*** software life cycle process that contains the activities of requirements analysis, design, coding, integration, testing, installation and support for acceptance of software products*. ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.5. ***4.*** activity of preparing documentation after it has been designed*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.14

3.842
development branch
1. branch where active product development takes place

NOTE A product build from the development branch will have the latest features, but will also likely be immature and unstable.

3.843
development plan
1. plan for guiding, implementing, and controlling the design and development of one or more products or services

cf. project plan

3.844
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*

NOTE It entails the specification, construction, testing, and delivery of a new application. During actualization, this project can be split up into a number of sub-projects. If these are carried out more or less in parallel, each being responsible for effectuating a certain sub-system of the total application, then each sub-project should 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.

3.845
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.*** a count that measures the functions provided to the users with the first installation of the software, delivered when the project is complete*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.846
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*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.10

cf. acceptance testing, operational testing, qualification testing

NOTE The criteria will vary based on the level of test being performed.

3.847
developmental baseline
1. the specifications that are in effect at a given time for a system under development*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.07

3.848
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 The developmental configuration is under the developer's control, and therefore is not called a baseline.

3.849
deviation
1. a departure from a specified requirement. ***2.*** a 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

3.850
device
1. a mechanism or piece of equipment designed to serve a purpose or perform a function

3.851
device interface task
1. a concurrent task that hides the characteristics of and interfaces to an external I/O device

3.852
DFD
1. data flow diagram

3.853
diagnostic
1. pertaining to the detection and isolation of faults or failures

EXAMPLE a diagnostic message, a diagnostic manual

3.854
diagnostic manual
1. a 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 Typically described are the diagnostic features of the system or component and the diagnostic tools available for its support.

3.855
diagonal microinstruction
1. a microinstruction capable of specifying a limited number of simultaneous operations needed to carry out a machine language instruction

cf. horizontal microinstruction, vertical microinstruction

NOTE Diagonal microinstructions fall, in size and functionality, between horizontal microinstructions and vertical microinstructions The designation 'diagonal' refers to this compromise rather than to any physical characteristic of the microinstruction.

3.856
diagram
1. an instantiation of the formal diagram structure that consists only of semantically and syntactically valid IDEF0 graphical statements*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.39

NOTE Each diagram is a single unit of an IDEF0 model that presents the top-level function that is the subject of the model (the A-0 context diagram), presents the context of the subject function (other context diagrams), or presents the details of a box (decomposition diagrams).

3.857
diagram boundary
1. an edge of a diagram in a diagram page*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.40

3.858
diagram feature
1. an element of a diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.41

NOTE Diagram features include boxes, arrow segments, arrow labels, ICOM codes, ICOM labels, model notes, and reader notes.

3.859
diagram feature reference
1. an expression that unambiguously identifies a diagram feature within an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.42

3.860
diagram number
1. that part of a diagram reference that corresponds to a diagram's parent function's node number. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.43

NOTE The diagram number refers to the diagram that details or decomposes the function designated by the same node number.

3.861
diagram page
1. a model page that contains a context diagram or a decomposition diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.44

3.862
diagram reference
1. an expression that unambiguously identifies a diagram and specifies the diagram's position in a specific model hierarchy. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.45

NOTE A diagram reference is composed of a model name, abbreviation and a diagram number.

3.863
diagram title
1. a verb or verb phrase that describes the overall function presented by a diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.46

NOTE The diagram title of a child diagram is the box name of its parent box.

3.864
dialog
1. the 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*

3.865
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

3.866
digit numeric character
1. character that represents a nonnegative integer. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.12. *Syn:* numeric character

EXAMPLE one of the characters 0, 1... F in the hexadecimal numeration system

3.867
digital
1. pertaining to data that consists of digits as well as to processes and functional units that use the data. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.04

3.868
digital computer
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.04

3.869
direct address
1. an address that identifies the storage location of an operand. *Syn:* one-level address

cf. immediate data, indirect address, n-level address, direct instruction

3.870
direct and manage project execution
1. [Process] the process of performing the work defined in the project management plan to achieve the project's objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.871
direct instruction
1. a computer instruction that contains the direct addresses of its operands

cf. immediate instruction, indirect instruction, absolute instruction, effective instruction

3.872
direct measure
1. a measure of an attribute that does not depend upon a measure of any other attribute*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.4

3.873
direct metric
1. a metric that does not depend upon a measure of any other attribute*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.4

3.874
direct metric value
1. a numerical target for a quality factor to be met in the final product*. IEEE Std 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

3.875
direct staff-hour
1. the amount of effort directly expended in creating a specific output product

3.876
directed graph
1. a graph (sense 2) in which direction is implied in the internode connections. *Syn:* digraph

cf. undirected graph

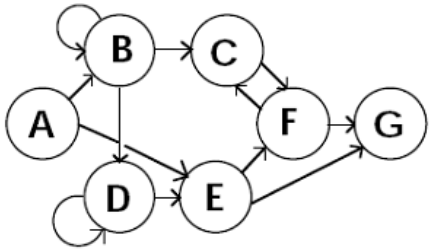


Figure 10 — Directed graph

3.877
director
1. member of the most senior governing body of an organization*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.5

NOTE Includes owners, board members, partners, senior executives or similar, and officers authorized by legislation or regulation

3.878
directory
1. a list of data items and information about those data items

3.879
disassemble
1. to translate an assembled computer program from its machine language version into a form that resembles, but may not be identical to, the original assembly language program

cf. assemble

3.880
disassembler
1. a software tool that disassembles computer programs

3.881
disclaimer
1. a notice that renounces or repudiates a legal claim or right

3.882
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.02

3.883
discrete data
1. data that arrives at specific time intervals

3.884
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

cf. apportioned effort

3.885
discrete type
1. a data type whose members can assume any of a set of distinct values

NOTE A discrete type may be an enumeration type or an integer type.

3.886
discrimination (threshold)
1. largest change in a stimulus that produces no detectable change in the response of a measuring instrument, the change in the stimulus taking place slowly and monotonically. *ISO/IEC TR 14143-3:2003, Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods.*3.4

NOTE The discrimination threshold may depend on, for example, noise (internal or external) or friction. It may also depend on the value of the stimulus. [International vocabulary of basic and general terms in metrology, 1993, definition 5.11]

3.887
discriminator
1. a property of a superclass, associated with a cluster of that superclass, whose value identifies to which subclass a specific instance belongs. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.51. ***2.*** an 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.51. *Syn:* category discriminator

NOTE Since the value of the discriminator (when a discriminator has been declared) is equivalent to the identity of the subclass to which the instance belongs, there is no requirement for a discriminator in identity-style modeling.

3.888
disk
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.52

3.889
distribute information
1. [Process] the process of making relevant information available. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.890
distributed data processing GSC
1. one of the 14 general system characteristics describing the degree to which the application transfers data among components of the application. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.891
distributed processing
1. information processing in which discrete components may be located in different places, and where communication between components may suffer delay or may fail. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.1

3.892
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 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 5

3.893
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

3.894
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

3.895
division of standards
1. division forms a family of standards serving complementary purposes*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.13

3.896
DL
1. Definition Language*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.897
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

3.898
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 TR 9294:2005, Information technology — Guidelines for the management of software documentation.*3.1. ***2.*** to create a document as in (1)*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.11. ***3.*** to add comments to a computer program. ***4.*** an item of documentation*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.10. ***5.*** a medium, and the information recorded on it, that generally has permanence and can be read by a person or a machine*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.11. ***6.*** information and its supporting medium*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.6. ***7.*** separately identified piece of documentation which could be part of a documentation set*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.15

EXAMPLE in software engineering: project plans, specifications, test plans, user manuals

NOTE Documents include both paper and electronic documents.

3.899
document control
1. the application of configuration management to the control of documents

3.900
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.17

3.901
documentation
1. a collection of documents on a given subject. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.12. ***2.*** any written or pictorial information describing, defining, specifying, reporting, or certifying activities, requirements, procedures, or results. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.12. ***3.*** the process of generating or revising a document. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.12. ***4.*** collection of related documents that are designed, written, produced and maintained. *ISO/IEC TR 9294:2005, Information technology — Guidelines for the management of software documentation.*3.2. ***5.*** information that explains how to use a software product. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.16. ***6.*** the management of documents, including identification, acquisition, processing, storage, and dissemination. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.12

EXAMPLE printed manuals, on-screen information, and stand-alone online help

NOTE can be provided as separate documentation or as embedded documentation or both

3.902
documentation development staff
1. staff involved in any phase of the planning, writing, editing and production of documentation. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.12

NOTE This includes authors, designers, illustrators and project management staff.

3.903
documentation plan
1. document which sets out the essential elements of the documentation project. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.13. **2.** plan identifying the documents to be produced during the system or software life cycle. *ISO/IEC 15289:2006, Systems and software engineering — Contents of systems and software life cycle information products*. 5.2

3.904
documentation tree
1. a diagram that depicts all of the documents for a given system and shows their relationships to one another

cf. specification tree

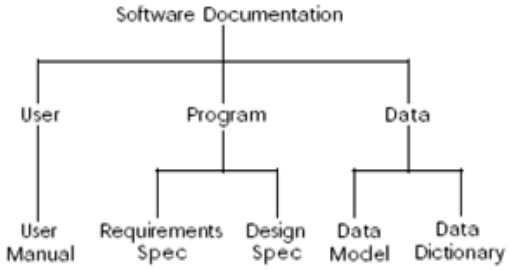


Figure 11 — Documentation tree

3.905
documenter
1. party preparing the documentation. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.14

3.906
domain
1. a distinct scope, within which common characteristics are exhibited, common rules observed, and over which a distribution transparency is preserved. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.5. ***2.*** a problem space. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes - Reuse Processes.*3.6

3.907
domain analysis
1. the analysis of systems within a domain to discover commonalities and differences among them. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.7. ***2.*** the 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 Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.7. ***3.*** the result of the domain analysis process. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.7

3.908
domain architecture
1. a generic, organizational structure or design for software systems in a domain. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.8

NOTE The domain architecture contains the designs that are intended to satisfy requirements specified in the domain model. The domain architecture documents design, whereas the domain model documents requirements. A domain architecture: 1) can be adapted to create designs for software systems within a domain, and 2) provides a framework for configuring assets within individual software systems. The term "architecture" has been deliberately redefined to more properly convey its meaning in the software reuse context.

3.909
domain definition
1. the process of determining the scope and boundaries of a domain. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.9

3.910
domain engineer
1. a party that performs domain engineering activities, including domain analysis, domain design, asset construction, and asset maintenance. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.10

3.911
domain engineering
1. a 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 Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.9

NOTE "Assets" may include requirements, designs, software code, or documentation. Domain engineering may include the following activities: domain definition, domain analysis, developing the domain architecture, and domain implementation.

3.912
domain expert
1. an individual who is intimately familiar with the domain and can provide detailed information to the domain engineers. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.12

3.913
domain model
1. a product of domain analysis that provides a representation of the requirements of the domain. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.13

NOTE The domain model identifies and describes the structure of data, flow of information, functions, constraints, and controls within the domain that are included in software systems in the domain. The domain model describes the commonalities and variabilities among requirements for software systems in the domain.

3.914
dot notation
1. a technique for naming that joins the name of a parent class to the name of a dependent class with the period character. *IEEE Std 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

3.915
down
1. pertaining to a system or component that is not operational or has been taken out of service

cf. up, busy, crash, idle

3.916
down time
1. the 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

3.917
download
1. to transfer programs or data from a computer to a connected computer with fewer resources. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.36

NOTE typically from a server to a personal computer

3.918
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

3.919
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

3.920
driver
1. a software module that invokes and, perhaps, controls and monitors the execution of one or more other software modules. ***2.*** a computer program that controls a peripheral device and, sometimes, reformats data for transfer to and from the device

cf. test driver

3.921
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

3.922
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

3.923
dumb terminal
1. user terminal that has no independent data processing capability*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.15. *Syn:* nonprogrammable terminal

3.924
dump
1. a display of some aspect of a computer program's execution state, usually the contents of internal storage or registers. ***2.*** a 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 Types include change dump, dynamic dump, memory dump, postmortem dump, selective dump, snapshot dump, static dump.

3.925
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) — Fourth Edition*

3.926
dyadic selective construct
1. an if-then-else construct in which processing is specified for both outcomes of the branch

cf. monadic selective construct

3.927
dynamic
1. pertaining to an event or process that occurs during computer program execution

cf. static

EXAMPLE dynamic analysis, dynamic binding

3.928
dynamic analysis
1. the process of evaluating a system or component based on its behavior during execution

cf. static analysis demonstration, testing

3.929
dynamic binding
1. binding performed during the execution of a computer program

cf. static binding

3.930
dynamic breakpoint
1. a 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

3.931
dynamic buffering
1. a buffering technique in which the buffer allocated to a computer program varies during program execution, based on current need

cf. simple buffering

3.932
dynamic dump
1. a dump that is produced during the execution of a computer program

cf. static dump, change dump, memory dump, postmortem dump, selective dump, snapshot dump

3.933
dynamic error
1. an error that is dependent on the time-varying nature of an input

cf. static error

3.934
dynamic invocation
1. constructing and issuing a request whose signature is possibly not known until run-time. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.2.6

3.935
dynamic model
1. a model that describes individual requests or patterns of requests among objects*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.53

cf. static model

3.936
dynamic relocation
1. relocation of a computer program during its execution

3.937
dynamic resource allocation
1. a computer resource allocation technique in which the resources assigned to a program vary during program execution, based on current need

3.938
dynamic restructuring
1. the process of restructuring a database, data structure, computer program, or set of system components during program execution

3.939
dynamic schema
1. a specification of the allowable state changes of one or more information objects, subject to the constraints of any invariant schemata*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*6.1.3

NOTE Behavior in an information system can be modeled as transitions from one static schema to another, i.e., reclassification of instances from one type to another. In the information language, a state change involving a set of objects can be regarded as an interaction between those objects. Not all of the objects involved in the interaction need change state; some of the objects may be involved in a read-only manner.

3.940
dynamic skeleton
1. an interface-independent kind of skeleton, used by servers to handle requests whose signatures are possibly not known until run-time*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.7

3.941
dynamic storage allocation
1. a 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

3.942
EAC
1. estimate at completion*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.943
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 (or the project) can finish, based on the schedule network logic, the data date, and any schedule constraints. Early finish dates can change as the project progresses and as changes are made to the project management plan*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.944
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 (or the project) can start, based on the schedule network logic, the data date, and any schedule constraints. Early start dates can change as the project progresses and as changes are made to the project management plan*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.945
early-failure period
1. the 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. *Syn:* burn-in period

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

3.946
earned value (EV)
1. the value of work performed expressed in terms of the approved budget assigned to that work for a schedule activity or work breakdown structure component*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* budgeted cost of work performed (BCWP)

3.947
earned value management (EVM)
1. a management methodology for integrating scope, schedule, and resources, and for objectively measuring project performance and progress. Performance is measured by determining the budgeted cost of work performed (i.e., earned value) and comparing it to the actual cost of work performed (i.e., actual cost). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.948
earned value technique (EVT)
1. [Technique] a specific technique for measuring the performance of work and used to establish the performance measurement baseline (PMB). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.949
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.*** a returned signal

3.950
ECP
1. engineering change proposal

3.951
EDI
1. electronic data interchange

3.952
edit
1. to modify the form or format of computer code, data, or documentation

EXAMPLE to insert, rearrange, or delete characters

3.953
EF
1. early finish date. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.954
effective address
1. the 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 If the specified address requires no modification, it is also the effective address.

3.955
effective full license
1. license rights for software which allow one full use of the software. *ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*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 An effective license consists of one or more underlying licenses.

3.956
effective instruction
1. the 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 If the specified instruction requires no modification, it is also the effective instruction.

3.957
effectiveness
1. producing the intended or desired result. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** the 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

3.958
effectiveness analysis
1. an analysis of how well a design solution will perform or operate given anticipated operational scenarios. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.10

3.959
effectiveness assessment
1. the evaluation of the design solution with respect to manufacturing, test, distribution, operations, support, training, environmental impact, cost-effectiveness, and life cycle cost. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.11

3.960
efferent
1. pertaining to a flow of data or control from a superordinate module to a subordinate module in a software system

cf. afferent

3.961
efficiency
1. the degree to which a system or component performs its designated functions with minimum consumption of resources. ***2.*** producing a result with a minimum of extraneous or redundant effort. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***3.*** 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

cf. execution efficiency, storage efficiency

3.962
effort
1. the number of labor units required to complete a schedule activity or work breakdown structure component. Usually expressed as staff hours, staff days, or staff weeks. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. duration

3.963
egoless programming
1. a software development technique based on the concept of team, rather than individual, responsibility for program development

NOTE Its purpose is to prevent individual programmers from identifying so closely with their work that objective evaluation is impaired.

3.964
EI
1. external 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*

3.965
EIF
1. 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*

3.966
electronic copy
1. computer disk or other computer-readable medium containing a file or files from which the document can be printed*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.15

3.967
electronic data interchange (EDI)
1. structured way of transmitting data held electronically from database to database, usually using telecommunications networks

3.968
electronic mail (Email)
1. correspondence in the form of messages transmitted over a computer network*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.17. *Syn:* e-mail

3.969
electronic publishing
1. the production of typeset-quality documents including text, graphics, and pictures with the assistance of a computer*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.15

3.970
element
1. a component of a system; may include equipment, a computer program, or a human*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.7

EXAMPLE documents, requirements specifications, test cases, source code, installation information, and read-me files

3.971
element type
1. a category or class of elements

3.972
elementary process
1. the smallest unit of activity that is meaningful to the user(s)*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6

3.973
ELSE-rule
1. the actions to be taken for all combinations of conditions not covered by the other rules in the table*. ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.5

NOTE The use of the ELSE-rule facility is optional.

3.974
embedded computer system
1. a 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

3.975
embedded documentation
1. documentation that is accessed as an integral part of software*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.18

cf. separate documentation

EXAMPLE pop-up help and help text on a screen

3.976
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

3.977
emergency maintenance
1. an unscheduled modification performed to temporarily keep a system operational pending corrective maintenance*. ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.3

NOTE Emergency maintenance is a part of corrective maintenance.

3.978
emulated user
1. the 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

3.979
emulation
1. a model that accepts the same inputs and produces the same outputs as a given system. ***2.*** the 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.02

cf. simulation

3.980
emulator
1. a device, computer program, or system that accepts the same inputs and produces the same outputs as a given system

cf. simulator

3.981
EMV
1. expected monetary value*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.982
en dash
1. dash the same width as a lower-case 'n'. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.16

3.983
enabling (a transition)
1. a 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 The determination of transition modes guarantees that the transition condition is satisfied.

3.984
enabling system
1. a system that complements a system-of-interest during its life cycle stages but does not necessarily contribute directly to its function during operation. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.11; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.9

NOTE 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.

3.985
enabling tokens
1. the 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

3.986
encapsulation
1. a 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.*** the concept that access to the names, meanings, and values of the responsibilities of a class is entirely separated from access to their realization. *IEEE Std 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

3.987
encoding
1. a definition of how the elements of a syntax are represented using an identified character set. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE Details of representation of the various terminal symbols and data types in the syntax's grammar are provided.

3.988
ENCODING.1
1. the primary encoding defined within the CDIF family of standards. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE The CDIF family of standards supports multiple transfer formats, each composed of a syntax and an encoding.

3.989
end item
1. an entity (hardware equipment, software equipment, data, facilities, material, services, and/or techniques) identified with an element of the SBS*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.12

3.990
end user
1. the person or persons who will ultimately be using the system for its intended purpose*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.* ***2.*** individual person who ultimately benefits from the outcomes of the system*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.14. ***3.*** the person who uses the software package*. ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.3. ***4.*** any person that communicates or interacts with the software at any time*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.5. *Syn:* end-user

3.991
endeavor
1. IBD development effort aimed at the delivery of some product or service through the application of a methodology*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.5. *Syn:* endeavour

EXAMPLE projects, programs and infrastructural duties

3.992
endeavor element
1. simple component of an endeavor*. ISO/IEC 24744:2007, 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 During the execution of an endeavor, developers create a number of endeavor elements, such as tasks, models, classes, documents.

3.993
endnotes
1. notes collected at the end of a chapter or document*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.17

3.994
end-user efficiency GSC
1. one of the 14 general system characteristics describing the degree of consideration for human factors and ease of use for the user of the application measured*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.995
engineering
1. the application of a systematic, disciplined, quantifiable approach to structures, machines, products, systems, or processes

3.996
engineering change
1. an 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

3.997
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

3.998
engineering interface reference
1. an identifier, in the context of an engineering interface reference management domain, for an engineering object interface that is available for distributed binding. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.16

NOTE An engineering interface reference is necessary to establish distributed bindings, and is distinct from the binding endpoint identifiers used by a basic engineering object for the purposes of interaction.

3.999
engineering interface reference management domain
1. a set of nodes forming a naming domain for the purpose of assigning engineering interface references. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.17

3.1000
engineering interface reference management policy
1. a set of permissions and prohibitions that govern the federation of engineering interface reference management domains. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.18

3.1001
engineering viewpoint
1. a 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.1.1.4

3.1002
enhancement
1. the modification of an existing application. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** 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*

3.1003
enhancement project
1. 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 This means that functionality can be added to, changed in, or deleted from an existing application.

3.1004
enhancement project function point count (EFP)
1. a count that measures the modifications to the existing application that add, change, or delete user functions delivered when the project is complete. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** a 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*

3.1005
enterprise
1. the organization that performs specified tasks*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.13

NOTE An organization may be involved in several enterprises and an enterprise may involve one or more organizations`.

3.1006
enterprise environmental factors
1. [Output/Input] any or all external environmental factors and internal organizational environmental factors that surround or influence the project's success. These factors are from any or all of the enterprises involved in the project, and include organizational culture and structure, infrastructure, existing resources, commercial databases, market conditions, and project management software*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1007
enterprise viewpoint
1. a viewpoint on an ODP system and its environment that focuses on the purpose, scope, and policies for that system*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.1.1.1

3.1008
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.*** an 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 15939:2007, Systems and software engineering — Measurement process.*3.9; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.15. ***5.*** the 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.55. ***6.*** a fundamental thing of relevance to the user, about which a collection of facts is kept*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.8

EXAMPLE a data item, program statement, or subprogram

3.1009
entity attribute
1. a named characteristic or property of a design entity*. IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions.*3.3

NOTE It provides a statement of fact about the entity.

3.1010
entity instance
1. one of a set of real or abstract things represented by an entity*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.56

NOTE Each instance of an entity can be specifically identified by the value of the attribute(s) participating in its primary key. [key style]

3.1011
entity subtype
1. a subdivision of an entity type*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. entity

NOTE A subtype inherits all the attributes and relationships of its parent entity type, and may have additional, unique attributes and relationships.

3.1012
entity-relationship (E-R) diagram
1. a diagram that depicts a set of real-world entities and the logical relationships among them. *Syn:* entity-relationship map

cf. data structure diagram

3.1013
entry (-type)
1. a data movement type that moves a data group from a user across the boundary into the functional process where it is required*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method*.3.7

NOTE In COSMIC-FFP, an Entry is considered to include certain associated data manipulations (e.g. validation of the entered data). A data manipulation is anything that happens to data other than a movement.

3.1014
entry criteria
1. states of being that must be present before an effort can begin successfully

cf. exit criteria

3.1015
entry field
1. area on a screen or in a window in which a user enters data*. ISO/IEC 26514, Systems and software engineering — requirements for designers and developers of user documentation.*4.19

3.1016
entry point
1. a point in a software module at which execution of the module can begin. *Syn:* entrance, entry

cf. exit, reentry point

3.1017
enumeration type
1. a 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

3.1018
environment
1. 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 Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections — Classification and Description.*3.6. ***2.*** the configuration(s) of hardware and software in which the software operates*. ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.8. ***3.*** the circumstances, objects, and conditions that surround a system to be built*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.9. ***4.*** the circumstances, objects, and

conditions that will influence the completed system. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications*.3.9. ***5.*** a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.57

3.1019
EO
1. external output. *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*

3.1020
epilog breakpoint
1. a breakpoint that is initiated upon exit from a given program or routine. *Syn:* postamble breakpoint

cf. prolog breakpoint, code breakpoint, data breakpoint, dynamic breakpoint, programmable breakpoint, static breakpoint

3.1021
EQ
1. 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*

3.1022
equivalent faults
1. two or more faults that result in the same failure mode

3.1023
E-R diagram
1. entity-relationship diagram

3.1024
ERA
1. Entity-Relationship-Attribute modeling. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.5.2

3.1025
ergonomics
1. the scientific discipline concerned with the understanding of the interactions among human and other elements of a system. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)*.12. ***2.*** the profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)*.12

3.1026
errata
1. severe service-disrupting bugs for which there is no known workaround

NOTE Fixes for such bugs can often be introduced on a frozen branch.

3.1027
error
1. a human action that produces an incorrect result, such as software containing a fault. ***2.*** an incorrect step, process, or data definition. ***3.*** an incorrect result. ***4.*** the difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition

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

3.1028
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*

3.1029
error model
1. in software evaluation, a model used to estimate or predict the number of remaining faults, required test time, and similar characteristics of a system. *Syn:* error prediction model

3.1030
error prediction
1. a quantitative statement about the expected number or nature of faults in a system or component

cf. error model, error seeding

3.1031
error processing
1. the process of detecting and responding to a program's errors

3.1032
error seeding
1. the 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. *Syn:* bug seeding, fault seeding

cf. indigenous error

3.1033
error tolerance
1. the ability of a system or component to continue normal operation despite the presence of erroneous inputs

cf. fault tolerance, robustness

3.1034
ES
1. early start date*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1035
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

3.1036
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.20

3.1037
establish and maintain
1. to formulate, document, and use [a policy or procedure] throughout an organization

NOTE This phrase means more than a combination of its component terms; it includes documentation and usage.

3.1038
estimate
1. [Output/Input] 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)*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. budget

3.1039
estimate activity durations
1. [Process] 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) — Fourth Edition*

3.1040
estimate activity resources
1. [Process] 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) — Fourth Edition*

3.1041
estimate at completion (EAC)
1. [Output/Input] the expected total cost of a schedule activity, a work breakdown structure component, or the project when the defined scope of work will be completed. The EAC may be calculated based on performance to date or estimated by the project team based on other factors, in which case it is often referred to as the latest revised estimate*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. earned value technique, estimate to complete

3.1042
estimate costs
1. [Process] 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) — Fourth Edition*

3.1043
estimate to complete (ETC)
1. [Output/Input] the expected cost needed to complete all the remaining work for a schedule activity, work breakdown structure component, or the project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. earned value technique, estimate at completion

3.1044
estimated function point count
1. a 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 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).

3.1045
ETC
1. estimate to complete*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1046
EV
1. earned value*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1047
evaluation
1. systematic determination of the extent to which an entity meets its specified criteria*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.12. ***2.*** an action that assesses the value of something*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.5.5

EXAMPLE the action by which an ODP system assigns a relative status to some thing according to estimation by the system. Value can be considered in terms of usefulness, importance, preference, acceptability, etc.; the evaluated target may be a credit rating, a system state, a potential behavior

3.1048
evaluation activity
1. assessment of a software product against identified and applicable quality characteristics performed using applicable techniques or methods*. ISO/IEC 25001:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Planning and management.*4.1

3.1049
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 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.2

3.1050
evaluation group
1. organization responsible for specifying the software quality requirements as well as managing and implementing the software quality evaluation activities through the provision of technology, tools, experiences, and management skills*. ISO/IEC 25001:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Planning and management.*4.2

3.1051
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 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.16

3.1052
evaluation module
1. a package of evaluation technology for a specific software quality characteristic or sub-characteristic*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.5

3.1053
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 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.3

3.1054
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

3.1055
evaluation report
1. a 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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.02. ***2.*** the 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

3.1056
evaluation requester
1. the person or organization that requests an evaluation. *ISO/IEC 14598-5:1998, Information technology — Software product evaluation — Part 5: Process for evaluators.*4.4

3.1057
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 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.4

3.1058
evaluation technology
1. techniques, processes, tools, measures and relevant technical information used for evaluation. *ISO/IEC 25001:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Planning and management.*4.3. *Syn:* technology used for evaluation

NOTE These include internal, external or quality in use measures or specific evaluation processes designed for developers, acquirers, or independent evaluators.

3.1059
evaluation tool
1. an 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 tool inspection data or spreadsheets to produce syntheses of measures

3.1060
evaluator
1. individual or organization that performs an evaluation. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.18. ***2.*** the 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

3.1061
event
1. occurrence of a particular set of circumstances. *ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.2. ***2.*** an external or internal stimulus used for synchronization purposes

NOTE 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. [ISO Guide 73:2002, definition 3.1.4] An event can be an external interrupt, a timer expiration, an internal signal, or an internal message.

3.1062
event history
1. an object representing significant actions. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*13.1.1.1

3.1063
event sequence analysis
1. performance analysis of the sequence of tasks that must be executed to service a given external event

3.1064
event sequence diagram
1. a diagram that identifies the sequence of tasks required to process an external event

3.1065
event synchronization
1. control of task activation by means of signals

NOTE Three types of event synchronization are possible: external interrupts, timer expiration, and internal signals from other tasks.

3.1066
event trace
1. a time-ordered description of each external input and the time at which it occurred

3.1067
event-sequencing logic
1. a description of how a task responds to each of its message or event inputs

NOTE in particular, what output is generated as a result of each input

3.1068
EVM
1. earned value management. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1069
exception
1. an event that causes suspension of normal program execution

NOTE Types include addressing exception, data exception, operation exception, overflow exception, protection exception, and underflow exception.

3.1070
exception handling
1. a programming language mechanism that passes error information by throwing and catching exceptions

3.1071
exclusive requirement
1. requirement of a normative document that must necessarily be fulfilled in order to comply with that document. *ISO/IEC 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.5 e

NOTE deprecated: mandatory requirement. [ISO/IEC Guide 2:2004]

3.1072
executable requirements specification
1. a software requirement specification that is represented in an executable requirements language

3.1073
executable source statement
1. source statement that directs the actions of the computer at run time

3.1074
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) — Fourth Edition*

3.1075
executing processes
1. [Process Group] those processes performed to complete the work defined in the project management plan to satisfy the project's objectives. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1076
execution efficiency
1. the degree to which a system or component performs its designated functions with minimum consumption of time

cf. execution time, storage efficiency

3.1077
execution time
1. the 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.*** the amount of elapsed time or processor time used in executing a computer program. *Syn:* run time, running time

NOTE Processor time is usually less than elapsed time because the processor may be idle (for example, awaiting needed computer resources) or employed on other tasks during the execution of a program.

3.1078
execution trace
1. a record of the sequence of instructions executed during the execution of a computer program. *Syn:* code trace, control flow trace

cf. retrospective trace, subroutine trace, symbolic trace, variable trace

NOTE Often takes the form of a list of code labels encountered as the program executes.

3.1079
existence constraint
1. a constraint stating that an instance of one entity cannot exist unless an instance of another related entity also exists. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.59

NOTE [key style]

3.1080 existence dependency
1. a constraint between two related entities indicating that no instance of one can exist without being related to an instance of the other. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.60

NOTE The following association types represent existence dependencies: identifying relationships, categorization structures and mandatory nonidentifying relationships. [key style]

3.1081
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*. ISO/IEC 14598-4:1999, Software engineering — Product evaluation — Part 4: Process for acquirers.*4.3

3.1082
exit
1. a point in a software module at which execution of the module can terminate. ***2.*** a data movement type that moves a data group from a functional process across the boundary to the user that requires it*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.8

cf. entry point, return

3.1083
exit criteria
1. states of being that must be present before an effort can end successfully

cf. entry criteria

3.1084
exit routine
1. a routine that receives control when a specified event, such as an error, occurs

3.1085
expandability
1. the degree of effort required to improve or modify software functions' efficiency

3.1086
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) — Fourth Edition*

3.1087
expert judgment
1. [Technique] 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.

3.1088
expert system (ES)
1. a 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 2382-1:1993, Information technology--Vocabulary--Part 1: Fundamental terms.*01.06.19

NOTE Some expert systems are able to improve their knowledge base and develop new inference rules based on their experience with previous problems.

3.1089
exploratory testing
1. simultaneous learning, test design, and test execution*. ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*5.3.1.2

NOTE The tests are not defined in advance in an established test plan, but are dynamically designed, executed, and modified.

3.1090
export process
1. the process of generating a transfer file from a source environment. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.1

3.1091
exporter
1. the agent of the export process. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.1

3.1092
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

3.1093
extendability
1. the ease with which a system or component can be modified to increase its storage or functional capacity. *Syn:* expandability, extensibility

cf. flexibility, maintainability

3.1094
extended entry table
1. a decision table where the conditions and actions are generally described but are incomplete. *ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.15

NOTE The specifications are completed by the values specified in the rules.

3.1095
extensional set
1. the set containing the currently existing instances of a class. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.61. *Syn:* current extent

NOTE The instances in the extensional set correspond to the database and data modeling notion of instance.

3.1096
external
1. an input information source or output information destination that is outside the scope of this standard and, therefore, may or may not exist. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*Annex E

cf. invocation, iteration, mapping

3.1097
external attribute
1. a measurable property of an entity which can only be derived with respect to how it relates to its environment. *ISO/IEC 14598-3:2000, Software engineering — Product evaluation — Part 3: Process for developers.*4.2

NOTE External attributes are those that relate to requirements (external properties of the software). External attributes can only be derived from the operational behavior of the system of which it is a part.

3.1098
external event
1. an event from an external object, typically an interrupt from an external I/O device

3.1099
external I/O device
1. a hardware input and/or output device that is outside the software system and part of the external environment

3.1100
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.*** an elementary process that processes data or control information that comes from outside the application's boundary*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. external inquiry, external output

NOTE The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system.

3.1101
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.*** an elementary process that sends data or control information outside the application boundary*. ISO/IEC 20926:2003 Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. external input, external output

NOTE 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.

3.1102
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.*** a user-identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6

cf. internal logical file

NOTE 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.

3.1103
external interface requirement
1. a 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

3.1104
external measure
1. an indirect measure of a product derived from measures of the behavior of the system of which it is a part*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.6

NOTE The system includes any associated hardware, software (either custom software or off-the-shelf software) and users. 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.

3.1105
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.*** an elementary process that sends data or control information outside the application's boundary*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. external input, external inquiry

NOTE The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information. The processing logic must contain at least one mathematical formula or calculation, or create derived data. An external output may also maintain one or more ILFs and/or alter the behavior of the system.

3.1106
external quality
1. the extent to which a product satisfies stated and implied needs when used under specified conditions*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.7

3.1107
external software quality
1. capability of a software product to enable the behavior of a system to satisfy stated and implied needs when the system is used under specified conditions*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.19

EXAMPLE The number of failures found during testing is an external software quality measure related to the number of faults present in the program. The two measures are not necessarily identical since testing may not find all faults, and a fault may give rise to apparently different failures in different circumstances

NOTE Attributes of the behavior can be verified and/or validated by executing the software product during testing and operation.

3.1108
extranet
1. a set of intranets connected for specific objectives, spanning multiple organizations*. IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.4

3.1109
facilitate change GSC
1. one of the 14 general system characteristics describing the degree to which the application has been developed for easy modification of processing logic or data structure*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1110
facility
1. physical means or equipment for facilitating the performance of an action*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.13

3.1111
facility
1. physical means or equipment for facilitating the performance of an action. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.10

EXAMPLE buildings, instruments, tools

3.1112
factoring
1. the process of decomposing a system into a hierarchy of modules. ***2.*** the process of removing a function from a module and placing it into a module of its own

cf. modular decomposition

3.1113
fail safe
1. pertaining to a system or component that automatically places itself in a safe operating mode in the event of a failure. *Syn:* fail-safe, failsafe

cf. fail soft, fault secure, fault tolerance

EXAMPLE a traffic light that reverts to blinking red in all directions when normal operation fails

3.1114
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

3.1115
failure
1. termination of the ability of a product to perform a required function or its inability to perform within previously specified limits. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.20. ***2.*** an event in which a system or system component does not perform a required function within specified limits

NOTE A failure may be produced when a fault is encountered.

3.1116
failure mode
1. the physical or functional manifestation of a failure

NOTE A system in failure mode may be characterized by slow operation, incorrect outputs, or complete termination of execution.

3.1117
failure mode and effect analysis (FMEA)
1. [Technique] 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) — Fourth Edition*

3.1118
failure rate
1. the ratio of the number of failures of a given category to a given unit of measure. *Syn:* failure ratio

EXAMPLE failures per unit of time, failures per number of transactions, failures per number of computer runs

3.1119
failure transparency
1. a distribution transparency which masks, from an object, the failure and possible recovery of other objects (or itself), to enable fault tolerance*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.2

3.1120
fast tracking
1. [Technique] a specific project schedule compression technique that changes network logic to overlap phases that would normally be done in sequence, such as the design phase and construction phase, or to perform schedule activities in parallel*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. schedule compression, crashing

3.1121
fatal error
1. an error that results in the complete inability of a system or component to function

3.1122
fault
1. a manifestation of an error in software. ***2.*** an incorrect step, process, or data definition in a computer program. ***3.*** a defect in a hardware device or component. *Syn:* bug

NOTE: A fault, if encountered, may cause a failure.

3.1123
fault dictionary
1. a list of faults in a system or component, and the tests that have been designed to detect them

3.1124
fault isolation
1. the ability of a subsystem to prevent a fault within the subsystem from causing consequential faults in other subsystems*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.5

3.1125
fault masking
1. a condition in which one fault prevents the detection of another

3.1126
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

3.1127
fault tolerance
1. the ability of a system or component to continue normal operation despite the presence of hardware or software faults. ***2.*** the number of faults a system or component can withstand before normal operation is impaired. ***3.*** 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

3.1128
fault-tolerant
1. pertaining to a system or component that is able to continue normal operation despite the presence of faults. *Syn:* fault tolerant

3.1129
FCA
1. functional configuration audit

3.1130
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

3.1131
FDT
1. formal description techniques. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1132
feasibility
1. the degree to which the requirements, design, or plans for a system or component can be implemented under existing constraints

3.1133
feasibility study
1. a study to identify and analyze a problem and its potential solutions in order to determine their viability, costs, and benefits. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.02

3.1134
feature
1. distinguishing characteristic of a system item. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.13

NOTE includes both functional and nonfunctional attributes such as performance and reusability

3.1135
feature branch
1. a branch created for developing a particular set of features

NOTE The branch is typically not released but is collapsed back at some point to its parent branch.

3.1136
feature freeze
1. a period during which no new features are added to a specific branch

NOTE allows the branch to stabilize for a release

3.1137
feature reference
1. an expression that unambiguously identifies a diagram feature in a diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.48

3.1138
fetch
1. to locate and load computer instructions or data from storage

cf. move, store

3.1139
FF
1. finish-to-finish. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1140
FFP
1. firm fixed price. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1141
field of application (of a specification)
1. the properties the environment of the ODP system must have for the specification of that system to be used. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.1.2

3.1142
fifth-generation language (5GL)
1. a 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

3.1143
figurative constant
1. a data name that is reserved for a specific constant in a programming language

cf. literal

EXAMPLE The data name THREE may be reserved to represent the value 3.

3.1144
file
1. a set of related records treated as a unit. ***2.*** for data functions, a logically related group of data, not the physical implementation of those groups of data. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***3.*** a named set of records stored or processed as a unit. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.08.06

EXAMPLE in stock control, a file could consist of a set of invoice records

3.1145
file type referenced (FTR)
1. an internal logical file read or maintained by a transactional function, or an external interface file read by a transactional function. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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*

3.1146
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*

3.1147
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) — Fourth Edition*

3.1148
finish-to-finish (FF)
1. the logical relationship where completion of work of the successor activity cannot finish until the completion of work of the predecessor activity. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. logical relationship

3.1149
finish-to-start (FS)
1. the logical relationship where initiation of work of the successor activity depends upon the completion of work of the predecessor activity. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. logical relationship

3.1150
finite state machine
1. a computational model consisting of a finite number of states and transitions between those states, possibly with accompanying actions

3.1151
firm-fixed-price (FFP) Contract
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) — Fourth Edition*

3.1152
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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.14. ***2.*** an ordered set of instructions and associated data stored in a way that is functionally independent of main storage, usually in a ROM. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.09

NOTE The software cannot be readily modified under program control.

3.1153
first input routine
1. those activities required to obtain the logical record, if any, to be processed first. *ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.4

3.1154
first normal form
1. result of a normalization process that transforms groups of data so they have a unique identifier, one or more attributes, and no repeating attributes. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1155
FiSMA
1. Finnish Software Measurement Association*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.10

NOTE a network of Finnish companies, which share interest in developing software measurement and/or software processes

3.1156
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) — Fourth Edition*

3.1157
flag
1. a 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

3.1158
flexibility
1. the 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. *Syn:* adaptability

cf. extendability, maintainability

3.1159
float
1. the amount of unscheduled time between sequential activities not on the critical path, which may be used to delay the completion of the earlier activity or advance the start of the later activity. *Syn:* slack

cf. free float, total float

3.1160
flow
1. an abstraction of a sequence of interactions, resulting in conveyance of information from a producer object to a consumer object*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1

NOTE A flow may 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.

3.1161
flowchart
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.06. ***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.*** a 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. *Syn:* flow chart, flow diagram

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

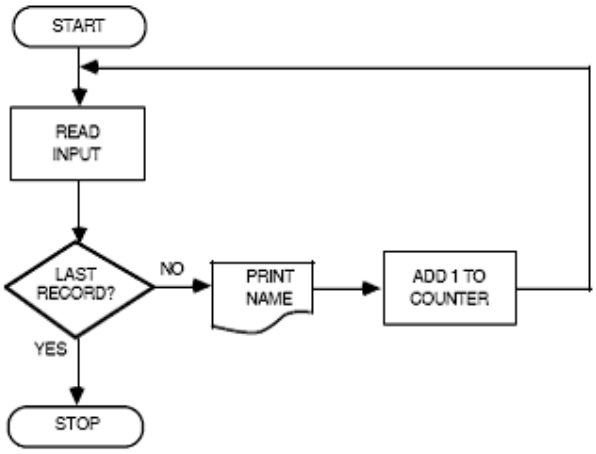


Figure 12 — Flowchart

3.1162
flowcharter
1. a software tool that accepts as input a design or code representation of a program and produces as output a flowchart of the program

3.1163
flowcharting
1. [Technique] 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) — Fourth Edition*

3.1164
FMEA
1. failure mode and effect analysis. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1165
foldout
1. single page wider than the rest, normally folded so that it does not protrude, that may be unfolded by the reader. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.18

cf. throwclear

3.1166
footer
1. material repeated at the bottom of each page. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.19

EXAMPLE page number

3.1167
footnote
1. text at the bottom of a page, usually in smaller type, which is referenced by means of a number or other device in the text on the same page. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.20

3.1168
For Exposition Only (FEO) page
1. a model page that contains pictorial and graphical information (in contrast to text) about a specific diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.51

NOTE Unlike a diagram, the contents of a For Exposition Only page (FEO page) need not comply with IDEF0 rules.

3.1169
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) — Fourth Edition*

3.1170
foreground
1. in job scheduling, the computing environment in which high-priority processes or those requiring user interaction are executed

cf. background, foreground processing

3.1171
foreground processing
1. the 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

3.1172
foreign key
1. an 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.62. ***2.*** data in an ILF or EIF that exists because the user requires a relationship with another ILF or EIF. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual. Syn:* migrated key

NOTE A foreign key results from the migration of the parent or generic entity's primary key through a generalization structure or a relationship. [key style]

3.1173
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

3.1174
formal design
1. the process of using a formal method for software design

3.1175
formal evaluation process
1. structured approach to evaluating alternative solutions against established criteria to determine a recommended solution to address an issue

3.1176
formal language
1. a language whose rules are explicitly established prior to its use

cf. natural language

EXAMPLE programming languages and mathematical languages

3.1177
formal parameter
1. a 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)

3.1178
formal qualification review (FQR)
1. the 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

3.1179
formal requirements language
1. an artificial language used to represent a software requirement. *Syn:* verifiable requirements language

NOTE The resulting formal requirements can be proven ""correct"" through proof-of-correctness methods.

3.1180
formal specification
1. a specification that is used to prove mathematically the validity of an implementation or to derive mathematically the implementation. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.04. ***2.*** a specification written in a formal notation, often for use in proof of correctness. ***3.*** a specification written and approved in accordance with established standards

3.1181
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

3.1182
formalization
1. the precise description of the semantics of a language in terms of a formal language such as first order logic. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.63

3.1183
forms design sheet
1. a 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

3.1184
forward pass
1. the calculation of the early start and early finish dates for the uncompleted portions of all network activities. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. schedule network analysis, backward pass

3.1185
forward recovery
1. the 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.*** a 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

3.1186
four-address instruction
1. a 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

3.1187
four-plus-one address instruction
1. a 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

3.1188
fourth-generation language (4GL)
1. a 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 Features typically include an integrated database management system, query language, report generator, and screen definition facility. Additional features may include a graphics generator, decision support function, financial modeling, spreadsheet capability, and statistical analysis functions.

3.1189
FPA
1. function point analysis*. 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*

3.1190
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*

3.1191
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*

3.1192
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*

3.1193
FP-EPA
1. fixed price with economic price adjustment*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1194
FPIF
1. fixed price incentive fee*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1195
F-profile
1. Format and presentation profile*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1196
FQR
1. formal qualification review

3.1197
framework
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.64. ***2.*** a partially completed software subsystem that can be extended by appropriately instantiating some specific plug-ins

3.1198
free float
1. the amount of time that a schedule activity can be delayed without delaying the early start date of any immediately following schedule activities*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. total float

3.1199
front matter
1. material that comes at the front of a book or manual, such as the title page and table of contents*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.21

3.1200
frozen branch
1. a branch where no development takes place, either in preparation for a release or because active development has ceased on it

3.1201
FS
1. finish-to-start*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1202
FSM
1. functional size measurement*. ISO/IEC 14143-1:2007, Information technology — Software measurement — Functional size measurement — Part 1: Definition of concepts.*4

3.1203
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

3.1204
FSMM
1. functional size measurement method. *ISO/IEC 14143-6, Information technology — Software measurement — Functional size measurement — Part 6: Guide for use of ISO/IEC 14143 series and related International Standards.*2

3.1205
FTR
1. file type referenced. *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*

3.1206
function
1. an 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* ***2.*** a software module that performs a specific action, is invoked by the appearance of its name in an expression, may receive input values, and returns a single value ***3.*** a task, action, or activity that must be accomplished to achieve a desired outcome. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.10. ***4.*** the features or capabilities of an application as seen by the user. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***5.*** a defined objective or characteristic action of a system or component. ***6.*** a 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.53. ***7.*** an aspect of the intended behavior of the system. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.6. ***8.*** a single-valued mapping. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.65. ***9.*** part of an application that provides facilities for users to carry out their tasks. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.21

3.1207
function name
1. an active verb or verb phrase that describes what is to be accomplished by a function. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.54

NOTE A box takes as its box name the function name of the function represented by the box.

3.1208
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.*** a measure which represents the functional size of application software. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1209
function point analysis (FPA)
1. a standard method for measuring software development and maintenance from the customer's point of view. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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.*** 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*

3.1210
function point count
1. the function point measurement of a particular application or project. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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*

3.1211
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*

3.1212
function type
1. the five basic information services provided to the user of an application and identified in function point analysis: external input, external output, external inquiry, internal logical file, and external interface file. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** 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*

3.1213
functional analysis
1. a systematic investigation of the functions of a real or planned system. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.06. ***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

3.1214
functional architecture
1. an 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. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.15. ***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

3.1215
functional baseline
1. in configuration management, the initial approved technical documentation for a configuration item. ***2.*** the initial configuration established at the end of the requirements definition phase

cf. allocated baseline, developmental configuration, product baseline

3.1216
functional cohesion
1. a 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

3.1217
functional complexity
1. a specific function type's complexity rating which has a value of low, average, or high. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE For data function types, the complexity is determined by the number of RETs and DETs. For transactional function types, the complexity is determined by the number of FTRs and DETs.

3.1218
functional configuration audit (FCA)
1. an 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

cf. configuration management, physical configuration audit

3.1219
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 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.

3.1220
functional decomposition
1. a 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

3.1221
functional design
1. the process of defining the working relationships among the components of a system. ***2.*** the result of the process in (1). ***3.*** the specification of the functions of the components of a system and of the working relationships among them. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.03.03

cf. architectural design

3.1222
functional domain
1. a 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

3.1223
functional domain categorization (FDC)
1. a process for identifying functional domains that conforms to the requirements of ISO/IEC TR 14143-5:2004, clause 53*. 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

3.1224
functional language
1. a programming language used to express programs as a sequence of functions and function calls

EXAMPLE: LISP

3.1225
functional manager
1. someone with management authority over an organizational unit within a functional organization. The manager of any group that actually makes a product or performs a service*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* line manager

3.1226
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) — Fourth Edition*

3.1227
functional process
1. an elementary component of a set of functional user requirements, comprising a unique, cohesive and independently executable set of data movements*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.9. ***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 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.11. *Syn:* transactional process

NOTE It is triggered by one or more Triggering events either directly, or indirectly via an 'actor'. It is complete when it has executed all that is required to be done in response to the Triggering event (-type).

3.1228
functional product
1. a 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)

3.1229
functional requirement
1. a statement that identifies what a product or process must accomplish to produce required behavior and/or results*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.* 3.1.16. ***2.*** a requirement that specifies a function that a system or system component must be able to perform

3.1230
functional service
1. service that must be implemented in the piece of software in order to fulfill functional user requirements*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.* A.12. ***2.*** base functional component (BFC)*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.6

3.1231
functional size
1. a 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. *Syn:* FS

3.1232
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

3.1233
functional specification
1. a document that specifies the functions that a system or component must perform

NOTE often part of a requirements specification

3.1234
functional testing
1. testing that ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions ***2.*** testing conducted to evaluate the compliance of a system or component with specified functional requirements. *Syn:* black-box testing

cf. structural testing

3.1235
functional unit
1. an entity of hardware or software, or both, capable of accomplishing a specified purpose*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.40

3.1236
functional user requirements (FUR)
1. 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

NOTE Functional User Requirements include but are not limited to: data transfer (for example, Input customer data, Send control signal); data transformation (for example, Calculate bank interest, Derive average temperature); data storage (for example, Store customer order, Record ambient temperature over time); data retrieval (for example, List current employees, Retrieve aircraft position). User Requirements that are not Functional User Requirements include but are not limited to: quality constraints (for example, usability, reliability, efficiency and portability); organizational constraints (for example, locations for operation, target hardware and compliance to standards); environmental constraints (for example, interoperability, security, privacy and safety); implementation constraints (for example, development language, delivery schedule).

3.1237
functionality
1. the capabilities of the various computational, user interface, input, output, data management, and other features provided by a product*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology-System Definition — Concept of Operation Document.* 3.1. ***2.*** capability of the software product to provide functions which meet stated and implied needs when the software is used under specified conditions. *ISO/IEC 9126-1:2001, Software engineering — Product quality — Part 1: Quality model.* 6.1.

NOTE This characteristic is concerned with what the software does to fulfil needs.

3.1238
function-oriented design
1. the partitioning of a design into subsystems and modules, with each one handling one or more functions

cf. object-oriented design, data-structure-oriented design

3.1239
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 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.* 4

3.1240
Gantt chart
1. [Tool] 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) — Fourth Edition*

3.1241
garbage collection
1. in computer resource management, a synonym for memory compaction (1)

3.1242
general system characteristics (GSCs)
1. IFPUG terminology for the technical complexity adjustment factors*. ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10. ***2.*** a set of 14 questions that evaluate the overall complexity of the application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1243
generality
1. the degree to which a system or component performs a broad range of functions

3.1244
generalization
1. a taxonomy in which instances of both entities represent the same real or abstract thing*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.66. *Syn:* categorization

NOTE One entity (the generic entity) represents the complete set of things and the other (category entity) represents a subtype or sub-classification of those things. The category entity may 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]

3.1245
generalization structure
1. a connection between a superclass and one of its more specific, immediate subclasses*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.69

3.1246
generalization taxonomy
1. a set of generalization structures with a common generic ancestor*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.70. *Syn:* generalization hierarchy, generalization network

NOTE In a generalization taxonomy every instance is fully described by one or more of the classes in the taxonomy. The structuring of classes as a generalization taxonomy determines the inheritance of responsibilities among classes.

3.1247
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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.66

NOTE Generalization is fundamentally different from a relationship, which may associate distinct instances.

3.1248
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 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*A

3.1249
generated address
1. an address that has been calculated during the execution of a computer program. *Syn:* synthetic address

cf. absolute address, effective address, indirect address, relative address

3.1250
generation
1. act of defining and describing a methodology from a particular metamodel*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.8

NOTE Generating a methodology includes explaining the structural position and semantics of each methodology element using the selected metamodel. Thus, what methodology elements are possible, and how they relate to each other, are constrained by such a metamodel. Usually, method engineers perform generation, yielding a complete and usable methodology.

3.1251
generic ancestor (of a class)
1. a superclass that is either an immediate superclass of the class or a generic ancestor of one of the superclasses of the class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.71

cf. reflexive ancestor

3.1252
generic entity
1. an entity whose instances are classified into one or more subtypes or subclassifications (category entities)*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.72. *Syn:* superclass, supertype

NOTE: [key style]

3.1253
generic practice
1. an activity that, when consistently performed, contributes to the achievement of a specific process attribute*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.22

3.1254
generic program unit
1. a 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

3.1255
GIF
1. General Interworking Framework*. ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*4

3.1256
GIOP
1. General Inter-ORB Protocol*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.1257
glass box
1. a system or component whose internal contents or implementation are known. ***2.*** pertaining to an approach that treats a system or component as in (1). *Syn:* white box

cf. black box

3.1258
global attribute
1. the 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.*6.1

3.1259
global compaction
1. in microprogramming, compaction in which microoperations may be moved beyond the boundaries of the single-entry, single-exit sequential blocks in which they occur

cf. local compaction

3.1260
global data
1. data that can be accessed by two or more non-nested modules of computer program without being explicitly passed as parameters between the modules. *Syn:* common data

cf. local data

3.1261
global variable
1. a 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

3.1262
glossary
1. the collection of the names and narrative descriptions of all terms that may be used for defined concepts within an environment. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.73. ***2.*** a set of definitions that includes arrow labels and box names used in an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.55

3.1263
glossary page
1. a model page that contains definitions for the arrow labels and box names in a specific diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.56

3.1264
go to
1. a computer program statement that causes a jump

cf. call, case, if-then-else branch

3.1265
goal
1. an intended outcome*. ISO/IEC TR 9126-4:2004, Software engineering — Product quality — Part 4: Quality in use metrics.*4.2. ***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

NOTE [ISO 9241-11]

3.1266
GOTS
1. Government-off-the-Shelf*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1267
Government-off-the-Shelf
1. software supplied by the government for reuse in another project. *Syn:* GOTS, Government-Off-The-Shelf, Government off the Shelf, Government Off The Shelf

cf. COTS

3.1268
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) — Fourth Edition*

3.1269
granularity
1. the depth or level of detail at which data is collected

3.1270
graph
1. a diagram that represents the variation of a variable in comparison with that of one or more other variables. ***2.*** a diagram or other representation consisting of a finite set of nodes and internode connections called edges or arcs

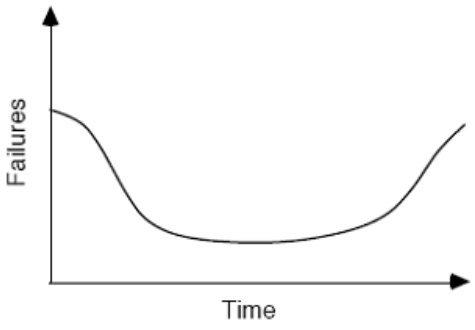


Figure 13 — Graph

EXAMPLE a graph showing a bathtub curve

3.1271
Grosch's law
1. a 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

3.1272
ground rules
1. list of acceptable and unacceptable behaviors adopted by a project team to improve working relationships, effectiveness, and communication

3.1273
GUI
1. Graphical User Interface*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1274
hacker
1. a technically sophisticated computer enthusiast*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.07.03. ***2.*** a technically sophisticated computer enthusiast who uses his or her knowledge and means to gain unauthorized access to protected resources*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.07.04

3.1275
halt
1. most commonly, a synonym for stop. ***2.*** less commonly, a synonym for pause

3.1276
hard copy
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.04

3.1277
hard failure
1. a failure that results in complete shutdown of a system

cf. soft failure

3.1278
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.010.07

cf. software

3.1279
hardware configuration item (HCI)
1. an aggregation of hardware that is designated for configuration management and treated as a single entity in the configuration management process ***2.*** a hardware entity that has been established as a configuration item

cf. software configuration item

NOTE An HCI exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the hardware has been established as a configuration item.

3.1280
hardware design language (HDL)
1. a language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a hardware design

cf. program design language

3.1281
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

3.1282
hardware monitor
1. a 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

3.1283
hazard
1. an intrinsic property or condition that has the potential to cause harm or damage*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*.3.1.11. ***2.*** a 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 Std 1012-2004 IEEE Standard for Software Verification and Validation*.3.1.11

3.1284
hazard identification
1. the process of recognizing that a hazard exists and defining its characteristics*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*.3.1.12

3.1285
HCI
1. human computer interface*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview* ***2.*** hardware configuration item. *Syn:* HWCI

3.1286
HDL
1. hardware design language

cf. design language

3.1287
HDTV
1. High Definition TV*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1288
head
1. the forefront of a branch, which contains the evolving versions of the source tree

NOTE A release coming out of head will have the newest features but will also likely be unstable.

3.1289
header
1. a 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*. ISO/IEC 15910:1999, Information technology — Software user documentation process*.4.22

3.1290
heading
1. text that identifies the topic that will be covered in the following text. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.23

3.1291
heavily used configuration GSC
1. one of the 14 general system characteristics describing the degree to which computer resource restrictions influenced the development of the application. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1292
heavyweight process
1. a process with its own memory and multiple threads of control

3.1293
help system
1. ancillary part of a program, or sometimes a separate program, that allows the user to view parts of the online documentation or help text on request. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.24

cf. online documentation system

3.1294
help text
1. text which is accessed by the user through the use of software, and which is automatically selected according to the context in which it is called up. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.25

NOTE Help text is context-sensitive.

3.1295
hidden
1. both private and protected. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.74

cf. public, private, protected

3.1296
hierarchical decomposition
1. a 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

3.1297
hierarchical modeling
1. a 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

3.1298
hierarchically consecutive
1. an unbroken unidirectional traversal of all nodes between two specified nodes in a tree. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.57

3.1299
hierarchy
1. a structure in which components are ranked into levels of subordination; each component has zero, one, or more subordinates; and no component has more than one superordinate component

cf. hierarchical decomposition, hierarchical modeling

3.1300
high level
1. general; abstract

3.1301
higher-level management
1. the 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 Such persons belong to a level of management in the organization above the immediate level responsible for the process.

3.1302
high-level design
1. the process of defining the high-level concepts that guide low-level design and implementation

cf. architecture

NOTE High-level design typically involves organizing a system into subprograms and specifying the interfaces between them.

3.1303
high-level net
1. an 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

3.1304
high-level Petri Net graph
1. a 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

3.1305
high-order language (HOL)
1. a 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. *Syn:* high-level language, high order language, higher order language, third-generation language

cf. assembly language, fifth-generation language, fourth-generation language, machine language

EXAMPLE Ada, COBOL, FORTRAN, ALGOL, PASCAL

3.1306
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) — Fourth Edition*

3.1307
HLL
1. **high-level language**
cf. high-order language

3.1308
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

3.1309
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

3.1310
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

3.1311
HMI
1. human-machine interface

cf. user interface

3.1312
HOL
1. high-order language

3.1313
homogeneous redundancy
1. in fault tolerance, realization of the same function with identical means

cf. diversity

EXAMPLE use of two identical processors

3.1314
horizontal microinstruction
1. a microinstruction that specifies a set of simultaneous operations needed to carry out a given machine language instruction

cf. diagonal microinstruction, vertical microinstruction

NOTE Horizontal microinstructions are relatively long, often 64 bits or more, and are called 'horizontal' because the set of simultaneous operations that they specify are written on a single line, rather than being listed sequentially down the page.

3.1315
host machine
1. the 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.*** a computer used to develop software intended for another computer. ***4.*** a computer used to emulate another computer

3.1316
hostile backout
1. a backout done without prior arrangement by a committer other than the one who introduced the original change

NOTE This is usually the opening shot in a commit war.

3.1317
housekeeping operation
1. a computer operation that establishes or reestablishes a set of initial conditions to facilitate the execution of a computer program. *Syn:* overhead operation

EXAMPLE initializing storage areas, clearing flags, rewinding tapes, opening and closing files

3.1318
human behavior
1. understanding of interactions among humans and other elements of a system with the intent to ensure well-being and systems performance. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.6. *Syn:* human behaviour

NOTE Human behavior includes culture, needs and aspirations of people as individuals and as groups.

3.1319
human resource plan
1. a document describing how roles and responsibilities, reporting relationships, and staffing management will be addressed and structured for the project. It is contained in or is a subsidiary plan of the project. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1320
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

3.1321
human systems engineering
1. the 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). *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.20

NOTE These activities include the definition and synthesis of manpower, personnel, training, human engineering, health hazards, and safety issues.

3.1322
HWCI
1. hardware configuration item

3.1323
hybrid computer
1. a computer that integrates analog computer components and digital computer components by interconnection of digital-to-analog converters and analog-to-digital converters. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.06

NOTE A hybrid computer may use or produce analog data and discrete data.

3.1324
hybrid coupling
1. a 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

3.1325
I/O
1. input/output

3.1326
I/O task-structuring criteria
1. a 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

3.1327
IBD
1. information-based domain. *ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*4.2

3.1328
ICOM code
1. an expression in one diagram that unambiguously identifies an arrow segment in another diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.58. *Syn:* arrow reference

NOTE An ICOM code is used to associate a boundary arrow of a child diagram with an arrow attached to an ancestral box.

3.1329
ICOM label
1. an 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.59

NOTE An ICOM label associates a boundary arrow of a child diagram with an arrow label of an arrow attached to an ancestral box.

3.1330
icon
1. graphic displayed on the screen that represents a function of the computer system. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.22

3.1331
ICS
1. Implementation Conformance Statement. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1332
ICT
1. information and communication technology. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.7

3.1333
ICWG
1. Interface Control Working Group. *ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5

NOTE Depending on the size and complexity of a project, can be a group of people, a single person or a function.

3.1334
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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.60

NOTE Concretely, a set of model pages that include at least an A-0 context diagram and an A0 decomposition diagram, a glossary or specific glossary pages, one or more text pages to accompany each diagram, and FEO pages and model pages of other types as needed.

3.1335
IDEF1X model
1. a set of one or more IDEF1X views, often represented as view diagrams that depict the underlying semantics of the views, along with definitions of the concepts used in the views*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.75

3.1336
identifier
1. the name, address, label, or distinguishing index of an object in a computer program. ***2.*** within an IDEF0 model, a model name, a box name, or an arrow label*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.61

3.1337
identifier dependency
1. a constraint between two related entities requiring the primary key in one (child entity) to contain the entire primary key of the other (parent entity)*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*3.1.76

NOTE Identifying relationships and categorization structures represent identifier dependencies. [key style]

3.1338
identify risks
1. [Process] 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) — Fourth Edition*

3.1339
identify stakeholders
1. [Process] the process of identifying all people or organizations impacted by the project, and documenting relevant information regarding their interests, involvement, and impact on project success*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1340
identifying relationship
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.79

cf. nonidentifying relationship [key style]

3.1341
identity
1. the inherent property of an instance that distinguishes it from all other instances*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.80

NOTE Identity is intrinsic to the instance and independent of the instance's property values or the classes to which the instance belongs.

3.1342
identity-style view
1. a view produced using the identity-style modeling constructs. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.81

3.1343
IDL
1. Interface Definition Language. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1344
idle
1. pertaining to a system or component that is operational and in service, but not in use

cf. busy, down, up

3.1345
idle time
1. the period of time during which a system or component is operational and in service, but not in use. *Syn:* standby time

cf. busy time, down time, set-up time, up time

3.1346
IEC
1. International Electrotechnical Commission. *ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5

3.1347
IFB
1. invitation for bid. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1348
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

3.1349
IFPUG
1. International Function Point Users Group. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE membership governed, non-profit organization committed to promoting and supporting function point analysis and other software measurement techniques. The IFPUG maintains the definition of the direct descendent of the Albrecht 1984 FPA method.

3.1350
if-then-else
1. a 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

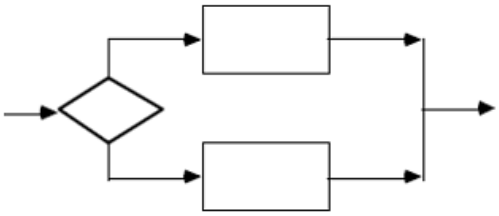


Figure 14 — If-then-else construct

3.1351
IIOP
1. Internet Inter-ORB Protocol*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.1352
IIOP-IOR
1. Internet Inter-ORB Protocol — Interoperable Object Reference*. ISO/IEC 14753:1999, Information technology — Open Distributed Processing — Interface references and binding.*4

3.1353
ILF
1. internal 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*

3.1354
illustration
1. *graphic element set apart from the main body of text and normally cited within the main text. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.23

NOTE used as the generic term for tables, figures, exhibits, screen captures, flow charts, diagrams, drawings, icons, and other graphic elements

3.1355
illustrative product
1. a 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)

3.1356
image processing
1. the use of a data processing system to create, scan, analyze, enhance, interpret, or display images*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.20

3.1357
immediate data
1. data contained in the address field of a computer instruction

cf. direct address, indirect address, n-level address, immediate instruction

3.1358
immediate instruction
1. a 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

3.1359
immutable class
1. a class for which the set of instances is fixed; its instances do not come and go over time*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.82

cf. mutable class, value class

3.1360
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 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.

3.1361
imperative construct
1. a sequence of one or more steps not involving branching or iteration

3.1362
implementable standard
1. a template for a technology object*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*9.1.1

3.1363
implementation
1. the process of translating a design into hardware components, software components, or both. ***2.*** the result of the process in (1). ***3.*** a 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 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.8. ***4.*** the installation and customization of packaged software. ***5.*** construction. ***6.*** the system development phase at the end of which the hardware, software and procedures of the system considered become operational*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.04.01. ***7.*** a process of instantiation whose validity can be subject to test*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*9.1.2. ***8.*** phase of development during which user documentation is created according to the design, tested, and revised*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.24

cf. coding

3.1364
implementation phase
1. period of time in the software life cycle during which a software product is created from design documentation and debugged

3.1365
implementation requirement
1. a 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

3.1366
implementer
1. organization that performs implementation tasks*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.15. *Syn:* developer

3.1367
implied addressing
1. a method of addressing in which the operation field of a computer instruction implies the address of the operands

cf. direct address, indirect address, relative address

EXAMPLE If a computer has only one accumulator, an instruction that refers to the accumulator needs no address information describing it. Types include one-ahead addressing and repetitive addressing.

3.1368
implied needs
1. needs that may not have been stated but are actual needs*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.23. ***2.*** needs that may not have been stated but are actual needs when the entity is used in particular conditions*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.10

NOTE Implied needs are real needs which may not have been documented.

3.1369
import process
1. the 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

3.1370
importer
1. the agent of the import process*. ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.1

3.1371
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) — Fourth Edition*

3.1372
incident
1. any event which is not part of the standard operation of a service and which causes or may cause an interruption to, or a reduction in, the quality of that service*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.7

cf. software test incident

NOTE may include request questions such as ""How do I?" calls

3.1373
incipient failure
1. a failure that is about to occur

3.1374
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

3.1375
incomplete process
1. process that is not performed or is performed only partially

NOTE One or more of the specific goals of the process are not satisfied.

3.1376
incremental compiler
1. a compiler that completes as much of the translation of each source statement as possible during the input or scanning of the source statement. *Syn:* conversational compiler, interactive compiler, online compiler

NOTE typically used for online computer program development and checkout

3.1377
incremental development
1. a 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. data structure-centered design, input-process-output, modular decomposition, object-oriented design, waterfall model

3.1378
incremental productivity
1. the productivity computed periodically during development

3.1379
independent
1. performed by an organization free from control by the supplier, developer, operator, or maintainer

3.1380
independent entity
1. an entity for which each instance can be uniquely identified without determining its relationship to another entity. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.84. *Syn:* identifier-independent entity

cf. dependent entity [key style]

3.1381
independent state class
1. a state class that is not a dependent state class. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.85

cf. dependent state class

3.1382
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

3.1383
indexed address
1. an 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

3.1384
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*

3.1385
indicator
1. measure that provides an estimate or evaluation of specified attributes derived from a model with respect to defined information needs*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*4.24; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.24. ***2.*** a measure that can be used to estimate or predict another measure*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.11. ***3.*** a 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

3.1386
indicator value
1. numerical or categorical result assigned to an indicator*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.11

3.1387
indigenous error
1. a computer program error that has not been purposely inserted as part of an error-seeding process

3.1388
indirect address
1. an address that identifies the storage location of another address. *Syn:* multilevel address

cf. direct address, immediate data, indirect instruction, n-level address

NOTE The designated storage location may contain the address of the desired operand or another indirect address; the chain of addresses eventually leads to the operand.

3.1389
indirect instruction
1. a computer instruction that contains indirect addresses for its operands

cf. direct instruction, immediate instruction, absolute instruction, effective instruction

3.1390
indirect measure
1. a measure of an attribute that is derived from measures of one or more other attributes*. ISO/IEC 14598-1:1999 Information technology — Software product evaluation — Part 1: General overview.*4.12

NOTE An external measure of an attribute of a computing system (such as the response time to user input) is an indirect measure of attributes of the software as the measure will be influenced by attributes of the computing environment as well as attributes of the software.

3.1391
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*

3.1392
infant mortality
1. the set of failures that occur during the early-failure period of a system or component

3.1393
influence diagram
1. [Tool] 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) — Fourth Edition*

3.1394
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

3.1395
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

3.1396
information
1. knowledge that is exchangeable amongst users about things, facts, concepts, and so on, in a universe of discourse*. ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.5. ***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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.01

NOTE Although information will necessarily have a representation form to make it communicable, it is the interpretation of this representation (the meaning) that is relevant in the first place.

3.1397
information analysis
1. a systematic investigation of information and its flow in a real or planned system*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.07

3.1398
information content
1. the 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

3.1399
information hiding
1. a 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

3.1400
information item
1. separately identifiable body of information that is produced and stored for human use during a system or software life cycle. *ISO/IEC 15289:2006, Systems and software engineering — Contents of systems and software life cycle information products (Documentation)*. 5.7

3.1401
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 15289:2006, Systems and software engineering — Contents of systems and software life cycle information products (Documentation)*. 5.8

3.1402
information item type
1. a group of information items consistent with a pre-arranged set of generic criteria. *ISO/IEC 15289:2006, Systems and software engineering — Contents of systems and software life cycle information products (Documentation)*. 5.8

EXAMPLE A "plan" is the information item type for all plans and "report" is the information item type for all reports.

3.1403
information management
1. in an information processing system, the functions of controlling the acquisition, analysis, retention, retrieval, and distribution of information. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.08.01

3.1404
information need
1. insight necessary to manage objectives, goals, risks, and problems. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.25; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.12

3.1405
information processing
1. the systematic performance of operations upon information, which includes data processing and may include operations such as data communication and office automation. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.05

NOTE The term information processing should not be used as a synonym for data processing.

3.1406
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

3.1407
information processing system
1. one or more data processing systems and devices, such as office and communication equipment, that perform information processing. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.21

3.1408
information product
1. one or more indicators and their associated interpretations that address an information need. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.13; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.26

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

3.1409
information retrieval (IR)
1. actions, methods, and procedures for obtaining information on a given subject from stored data*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.03

3.1410
information system
1. an information processing system, together with associated organizational resources such as human, technical, and financial resources, that provides and distributes information*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.22

3.1411
information system needs
1. needs that can be specified as quality requirements by external measures and sometimes by internal measures*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.27

3.1412
information technology
1. resources required to acquire, process, store and disseminate information*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.7. *Syn:* Information Technology, IT

NOTE includes Communication Technology (CT) and the composite term Information and Communication Technology (ICT)

3.1413
information viewpoint
1. a viewpoint on an ODP system and its environment that focuses on the semantics of information and information processing*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.1.1.2

3.1414
Information-based domain (IBD)
1. realm of activity for which information is the most valuable asset*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.1

NOTE Information creation, manipulation, and dissemination are the most important activities within information-based domains. Typical information-based domains are software and systems engineering, business process reengineering, and knowledge management.

3.1415
inheritance
1. a semantic notion by which the responsibilities (properties and constraints) of a subclass are considered to include the responsibilities of a superclass, in addition to its own, specifically declared responsibilities*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.86

3.1416
inherited attribute
1. an attribute that is a characteristic of a class by virtue of being an attribute of a generic ancestor*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.87. ***2.*** an 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.87

3.1417
inherited error
1. an error carried forward from a previous step in a sequential process

3.1418
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*

3.1419
initial marking (of the net)
1. the 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

3.1420
initial marking of a place
1. a 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

3.1421
initial program loader
1. a bootstrap loader used to load that part of an operating system needed to load the remainder of the operating system

3.1422
initialization section
1. an optional list of unconditional actions to be executed sequentially before the first condition is examined*. ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.13

NOTE It may be written in the row which follows that of the table heading.

3.1423
initialize
1. to set a variable, register, or other storage location to a starting value

cf. clear, reset

3.1424
initiating event
1. an event that can lead to a threat*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.7

3.1425
initiating processes
1. [Process Group] those processes performed to define a new project or a new phase by obtaining authorization to start the project or phase*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1426
initiator
1. person or organization that has both the ability and authority to start a project

3.1427
inline code
1. a sequence of computer instructions that is physically contiguous with the instructions that logically precede and follow it

3.1428
inner cardinality
1. the 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

3.1429
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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.62. ***7.*** [Process Input] 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) — Fourth Edition.* ***8.*** data entered into an information processing system or any of its parts for storage or processing*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.30. ***9.*** the process of entering data into an information processing system or any of its parts for storage or processing*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.31

3.1430
input arc (of a transition)
1. an 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

3.1431
input arrow
1. an arrow or arrow segment that expresses IDEF0 input*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0.*2.1.63

NOTE That is, an object type set whose instances are transformed by a function into output. The arrowhead of an input arrow is attached to the left side of a box.

3.1432
input assertion
1. a 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

3.1433
input loopback
1. loopback of output from one function to be input for another function in the same diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.64

3.1434
input place (of a transition)
1. a 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

3.1435
input primitive
1. the effort to develop software products, expressed in units of staff-hours

3.1436
input routine
1. those activities required to obtain the logical record to be processed next*. ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.5

NOTE If there are no more logical records to be processed, the end-of-input condition becomes true.

3.1437
input-process-output
1. a 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

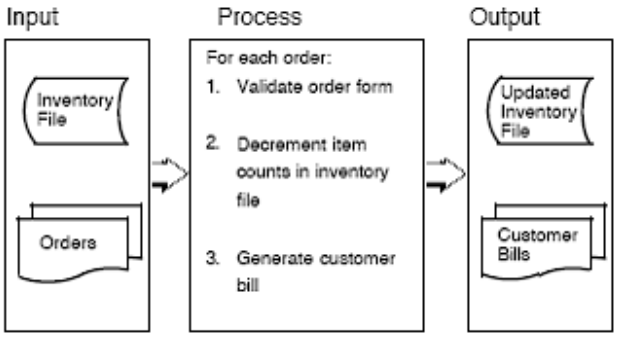


Figure 15 — Input-process-output chart

NOTE A refinement called hierarchical input-process-output identifies the steps, inputs, and outputs at both general and detailed levels of detail.

3.1438
input-process-output (IPO) chart
1. a diagram of a software system or module, consisting of a rectangle on the left listing inputs, a rectangle in the center listing processing steps, a rectangle on the right listing outputs, and arrows connecting inputs to processing steps and processing steps to outputs

cf. block diagram, box diagram, bubble chart, flowchart, graph, structure chart

3.1439
inspection
1. a visual examination of a software product to detect and identify software anomalies, including errors and deviations from standards and specifications*. IEEE Std 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.*** [Technique] 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) — Fourth Edition*

NOTE Inspections are peer examinations led by impartial facilitators who are trained in inspection techniques. Determination of remedial or investigative action for an anomaly is a mandatory element of a software inspection, although the solution should not be determined in the inspection meeting. Types include code inspection; design inspection.

3.1440
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

3.1441
installation ease GSC
1. one of the 14 general system characteristics describing the degree to which conversion from previous environments influenced the development of the application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1442
installation manual
1. a 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

3.1443
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

3.1444
instance
1. a discrete, bounded thing with an intrinsic, immutable, and unique identity*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.89. ***2.*** an individual occurrence of a type*. ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2. ***3.*** the mapping of an activity that processes all of its input information and generates all of its output information*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*3.1.5

3.1445
instance-level attribute
1. a mapping from the instances of a class to the instances of a value class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.90

3.1446
instance-level operation
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.91

3.1447
instance-level responsibility
1. a responsibility that applies to each instance of the class individually*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.92

cf. class-level responsibility

3.1448
instantiation
1. the process of substituting specific data, instructions, or both into a generic program unit to make it usable in a computer program

3.1449
institutionalization
1. ingrained way of doing business that an organization follows routinely as part of its corporate culture

3.1450
instruction counter
1. a register that indicates the location of the next computer instruction to be executed. *Syn:* program counter

3.1451
instruction cycle
1. the process of fetching a computer instruction from memory and executing it

cf. instruction time

3.1452
instruction format
1. the number and arrangement of fields in a computer instruction

cf. address field, address format, operation field

3.1453
instruction length
1. the number of words, bytes, or bits needed to store a computer instruction

cf. instruction format

3.1454
instruction modifier
1. a word or part of a word used to alter a computer instruction

3.1455
instruction set
1. the complete set of instructions recognized by a given computer or provided by a given programming language. *Syn:* instruction repertoire

3.1456
instruction time
1. the time it takes a computer to fetch an instruction from memory and execute it

cf. instruction cycle

3.1457
instructional mode
1. usage mode that is intended to teach the use of software in performing tasks*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.25

3.1458
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

3.1459
instrumentation
1. devices or instructions installed or inserted into hardware or software to monitor the operation of a system or component

3.1460
integer type
1. a 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

3.1461
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

3.1462
integrated circuit (IC)
1. a small piece of semiconductive material that contains interconnected electronic elements*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.10. *Syn:* microchip, chip

3.1463
integrated repository
1. a repository for storing all information pertinent to the sep to include all data, schema, models, tools, technical management decisions, process analysis information, requirement changes, process and product metrics, and trade-offs*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.18

3.1464
integrated team
1. group of people with complementary skills and expertise who are committed to delivering specified work products in timely collaboration

NOTE Integrated team members provide skills and advocacy appropriate to all phases of the work products' life and are collectively responsible for delivering work products as specified. An integrated team should include empowered representatives from organizations, disciplines, and functions that have a stake in the success of the work products.

3.1465
integration
1. the process of combining software components, hardware components, or both into an overall system

3.1466
integration test
1. the progressive linking and testing of programs or modules in order to ensure their proper functioning in the complete system*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.06. *Syn:* integration testing

3.1467
integration testing
1. testing in which software components, hardware components, or both are combined and tested to evaluate the interaction among them*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.14; *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.14

NOTE commonly used for both the integration of components and the integration of entire systems

3.1468
integrity
1. the degree to which a system or component prevents unauthorized access to, or modification of, computer programs or data

3.1469
integrity assurance authority
1. the independent person or organization responsible for assessment of compliance with the integrity requirements*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.8

3.1470
integrity level
1. value representing project-unique characteristics that define the importance of the software to the user*. IEEE Std 1012-2004 IEEE Standard for 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*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.15. ***3.*** symbolic value representing this degree of compliance within an integrity level scheme*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.15

NOTE "characteristics" such as software complexity, criticality, risk, safety level, security level, desired performance, reliability, or cost

3.1471
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*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.16

3.1472
intellectual property
1. an output of creative human thought process that has some intellectual or informational value

NOTE Intellectual property can be protected by patents, copyrights, trademarks, or trade secrets.

3.1473
interaction group
1. a subset of the objects participating in a binding managed by the group function*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*13.4.1.1

3.1474
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

3.1475
interactive language
1. a 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 The system provides questions, forms, and so on, to aid the user in expressing the results to be achieved.

3.1476
interconnection
1. an association between a computing system tool and something in the environment that affects both endpoints, though not necessarily in the same way*. IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnection — Classification and Description.*3.7

3.1477
interconnection feature
1. a property by which members of an interconnection perspective are characterized*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.6

3.1478
interconnection group
1. a collection of interconnections to a CASE tool that have a common kind of endpoint in the environment*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.7

3.1479
interconnection perspective
1. a subset of interconnections that share common features in an interconnection group*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.8

3.1480
interface
1. a shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical signal exchanges, and other characteristics. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.38. ***2.*** a 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.*** the declaration of the meaning and the signature for a property or constraint. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.93. ***6.*** a shared boundary across which information is passed. ***7.*** a task's external specification

3.1481
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

3.1482
interface design document (IDD)
1. a description of the architecture and design of interfaces between system and components

cf. interface requirements specification (IRS)

NOTE These descriptions include control algorithms, protocols, data contents and formats, and performance.

3.1483
interface repository
1. a storage place for interface information. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.9

3.1484
interface requirement
1. a 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

3.1485
interface requirements specification (IRS)
1. documentation that specifies requirements for interfaces between or among systems and components. *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.17; *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.17

cf. interface specification, interface design document

NOTE These requirements include constraints on formats and timing.

3.1486
interface role
1. a role of a community identifying behavior which takes place with the participation of objects that are not members of that community. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.3.4

3.1487
interface specification
1. the description of essential functional, performance, and design requirements and constraints at a common boundary between two or more system elements*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.19. ***2.*** a document that specifies the interface characteristics of an existing or planned system or component

cf. interface requirements specification

NOTE This includes interfaces between humans and hardware or software, as well as interfaces between humans themselves.

3.1488
interface task
1. a task that is part of the application, which interfaces to the external environment

3.1489
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

3.1490
interface type
1. a type satisfied by any object that satisfies a particular interface*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.18

3.1491
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 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.

3.1492
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

3.1493
intermediate software product
1. product of the software development process that is used as input to another stage of the software development process*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.28

EXAMPLE static and dynamic models, other documents and source code

3.1494
intermediate software product needs
1. needs that can be specified as quality requirements by internal measures*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.29

3.1495
intermittent fault
1. a temporary or unpredictable fault in a component

cf. random failure, transient error

3.1496
internal arrow
1. an arrow connected at both ends (source and use) to a box in a diagram*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.65

cf. boundary arrow

3.1497
internal attribute
1. a measurable property of an entity which can be derived purely in terms of the entity itself*. ISO/IEC 14598-3:2000, Software engineering — Product evaluation — Part 3: Process for developers.*4.3

NOTE Internal attributes are those that relate to the internal organization of the software and its development.

3.1498
internal event
1. a means of synchronization between two tasks

3.1499
internal logical file (ILF)
1. 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* ***2.*** a user-identifiable group of logically related data or control information maintained within the boundary of the application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6

cf. external interface file

NOTE The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted.

3.1500
internal measure
1. a measure of the product itself, either direct or indirect*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.14

NOTE The number of lines of code, complexity measures, the number of faults found in a walk through and the Fog Index are all internal measures made on the product itself.

3.1501
internal quality
1. the 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

3.1502
internal software quality
1. capability of a set of static attributes of a software product to satisfy stated and implied needs when the software product is used under specified conditions*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.30

EXAMPLE The number of lines of code, complexity measures and the number of faults found in a walkthrough are all internal software quality measures made on the product itself

NOTE Static attributes include those that relate to the software architecture, structure and its components. Static attributes can be verified by review, inspection and/or automated tools.

3.1503
internal task-structuring criteria
1. a category of the task-structuring criteria addressing how internal objects are mapped to internal tasks and when an internal task is activated

3.1504
internationalization
1. process of developing information so that it is suitable for an international audience*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.26

cf. localization

3.1505
interoperability
1. the ability of two or more systems or components to exchange information and to use the information that has been exchanged ***2.*** the ability for two or more ORBs to cooperate to deliver requests to the proper object*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.19. ***3.*** the 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.47

cf. compatibility

3.1506
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

3.1507
interpret
1. to translate and execute each statement or construct of a computer program before translating and executing the next

cf. assemble, compile

3.1508
interpreter
1. a computer program that translates and executes each statement or construct of a computer program before translating and executing the next

cf. assembler, compiler

3.1509
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

3.1510
interrogation
1. an 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, 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.4

NOTE In interrogations, invocations and terminations are always paired. Announcements do not have terminations. Thus there is no possibility of an operation consisting of an invocation followed by a sequence of associated terminations.

3.1511
interrupt
1. the 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. *Syn:* interruption

cf. interrupt latency, interrupt mask, interrupt priority, interrupt service routine, priority interrupt

3.1512
interrupt latency
1. the delay between a computer system's receipt of an interrupt request and its handling of the request

cf. interrupt priority

3.1513
interrupt mask
1. a mask used to enable or disable interrupts by retaining or suppressing bits that represent interrupt requests

3.1514
interrupt priority
1. the importance assigned to a given interrupt request

NOTE This importance determines whether the request will cause suspension of the current process and, if there are several outstanding interrupt requests, which will be handled first.

3.1515
interrupt request
1. a signal or other input requesting that the currently executing process be suspended to permit performance of another process

3.1516
interrupt service routine
1. a 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

3.1517
interval scale
1. scale in which the measurement values have equal distances corresponding to equal quantities of the attribute

EXAMPLE Cyclomatic complexity has the minimum value of one, but each increment represents an additional path. The value of zero is not possible

3.1518
Intranet
1. a managed network operating strictly within an organization. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.5

NOTE More than one Intranet may exist within an organization, these may be isolated.

3.1519
intrinsic
1. the specification that a property is total (i.e., mandatory), single-valued, and constant. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.94

3.1520
intrinsic relationship
1. a 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 Std 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 Such a relationship is considered to be an integral part of the essence of the dependent class.

3.1521
invariant
1. an assertion that should always be true for a specified segment or at a specified point of a computer program

3.1522
invariant schema
1. a set of predicates on one or more information objects which must always be true. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*6.1.1

NOTE The predicates constrain the possible states and state changes of the objects to which they apply. Thus, an invariant schema is the specification of the types of one or more information objects that will always be satisfied by whatever behavior the objects might exhibit.

3.1523
inverse engineering
1. the process of obtaining a high-level representation of the software from the source code

cf. reverse engineering

NOTE Inverse engineering provides a more abstract view of the system with the intent of recapturing design and requirements information.

3.1524
investment
1. allocation of human, capital, and other resources to achieve defined objectives and other benefits. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.8

3.1525
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) — Fourth Edition*

3.1526
invocation
1. the mapping of a parallel initiation of activities of an integral activity group that perform a distinct function and return to the initiating activity. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*3.1.7

cf. instance, iteration, mapping

3.1527
invocation deliver
1. a 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

3.1528
invocation submit
1. a 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

3.1529
IOR
1. Interoperable Object Reference. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.1530
IPO chart
1. input-process-output chart

3.1531
IPSE
1. integrated programming support environment

cf. programming support environment

3.1532
IRS
1. interface requirements specification. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1533
ISO
1. International Organization for Standardization. *ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5

3.1534
issue
1. a 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) — Fourth Edition*

cf. problem report

NOTE The record of an issue includes its identifier and brief description, and often identifies the environment associated with it, its status, severity, priority, and resolution, as well as dependencies, details on replicating or solving a problem, the persons associated with it, attachments, and its change history.

3.1535
IT
1. Information Technology. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1536
item
1. an entity such as a part, component, subsystem, equipment or system that can be individually considered An item may consist of hardware, software or both. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.1

3.1537
item of documentation
1. information designed for a specific audience for a specific purpose, and using a specific medium (e.g., book, disk, quick-reference card, video) of a particular format. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.26

3.1538
iteration
1. the process of performing a sequence of steps repeatedly. ***2.*** a single execution of the sequence of steps in (1)

cf. instance, invocation, mapping

NOTE One or more iterations comprise an instance.

3.1539
IV&V
1. independent verification and validation

3.1540
IXIT
1. Implementation Extra Information for Testing. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1541
JCL
1. job control language

3.1542
job
1. a user-defined unit of work that is to be accomplished by a computer

cf. job control language, job step, job stream

EXAMPLE the compilation, loading, and execution of a computer program

3.1543
job control language (JCL)
1. a language used to identify a sequence of jobs, describe their requirements to an operating system, and control their execution

3.1544
job function
1. a 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

3.1545
job step
1. a user-defined portion of a job, explicitly identified by a job control statement

NOTE: A job consists of one or more job steps.

3.1546
job stream
1. a sequence of programs or jobs set up so that a computer can proceed from one to the next without the need for operator intervention. *Syn:* run stream

3.1547
join
1. a 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.66

NOTE may denote bundling of arrows, meaning the inclusion of multiple object types within an object type set

3.1548
jump
1. to depart from the implicit or declared order in which computer program statements are being executed. ***2.*** a program statement that causes a departure as in (1). ***3.*** the departure described in (1). *Syn:* transfer

3.1549
junction
1. a point at which either a root arrow segment divides into branching arrow segments or arrow segments join into a root arrow segment. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.67

3.1550
kernel
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. *Syn:* nucleus, resident control program

3.1551
kernel entity
1. a classification used for a meta-entity whose instances can exist without the occurrences of other meta-entities. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

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.

3.1552
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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.96

NOTE [key style]

3.1553
key-style view
1. a view that represents the structure and semantics of data within an enterprise. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.97

NOTE That is, data (information) models.

3.1554
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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.98

3.1555
knowledge base (K-base)
1. a database that contains inference rules and information about human experience and expertise in a domain*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.18

NOTE In self-improving systems, the knowledge base additionally contains information resulting from the solution of previously encountered problems

3.1556
KOPS
1. kilo-operations per second; that is, thousands of operations per second

cf. MFLOPS, MIPS

NOTE a measure of computer processing speed

3.1557
label
1. a 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.*** a word or phrase that is attached to or part of a model graphic*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.99

3.1558
lag
1. [Technique] a modification of a logical relationship that directs a delay in the successor activity. For example, in a finish-to-start dependency with a ten-day lag, the successor activity cannot start until ten days after the predecessor activity has finished*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1559
language
1. a systematic means of communicating ideas by the use of conventionalized signs, sounds, gestures, or marks and rules for the formation of admissible expressions ***2.*** a means of communication, with syntax and semantics, consisting of a set of representations, conventions, and associated rules used to convey information

3.1560
language binding
1. the means and conventions by which a programmer writing in a specific programming language accesses ORB capabilities*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.2.20. *Syn:* language mapping

3.1561
language processor
1. a computer program that translates, interprets, or performs other tasks required to process statements expressed in a given language

cf. assembler, compiler, interpreter, translator

3.1562
language standard
1. a standard that describes the characteristics of a language used to describe a requirements specification, a design, or test data

3.1563
late finish date (LF)
1. in the critical path method, the latest possible point in time that a schedule activity may be completed based upon the schedule network logic, the project completion date, and any constraints assigned to the schedule activities without violating a schedule constraint or delaying the project completion date. The late finish dates are determined during the backward pass calculation of the project schedule network*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1564
late start date (LS)
1. in the critical path method, the latest possible point in time that a schedule activity may begin based upon the schedule network logic, the project completion date, and any constraints assigned to the schedule activities without violating a schedule constraint or delaying the project completion date. The late start dates are determined during the backward pass calculation of the project schedule network*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1565
latency
1. the 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

3.1566
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

3.1567
layer
1. the result of the functional partitioning of the software environment such that all included functional processes perform at the same level of abstraction*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.13

NOTE In a multi-layer software environment, software in one layer exchanges data with software in another layer through their respective functional processes. These interactions are hierarchical in nature; when considered in pairs, one layer is a "client" to the other. A "client" layer uses the functional services provided by other subordinate layers. Software items in the same layer can also exchange data. This type of data exchange is usually called "peer-to-peer" data exchange.

3.1568
layout
1. the physical organization of source code including the use of white space, grouping, blank lines, alignment, indentation, and parentheses

3.1569
layout chart
1. a 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

3.1570
lead
1. [Technique] a modification of a logical relationship that allows an acceleration of the successor activity. For example, in a finish-to-start dependency with a ten-day lead, the successor activity can start ten days before the predecessor activity has finished. A negative lead is equivalent to a positive lag*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. lag

3.1571
leading decision
1. a loop control that is executed before the loop body

cf. trailing decision, WHILE

3.1572
leaf diagram
1. a diagram that has no descendent diagrams*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.68

NOTE that is, a diagram that does not contain any function that has been decomposed

3.1573
leaf node
1. a function that is not decomposed*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.69

NOTE A box that represents a leaf node does not have a box detail reference.

3.1574
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) — Fourth Edition*

3.1575
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) — Fourth Edition*

3.1576
level
1. a designation of the coverage and detail of a view*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.100. ***2.*** the position in the hierarchy of control field elements, where a logical record contains more than one control field element*. ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.7

NOTE There are multiple levels of view; each is intended to be distinct, specified in terms of the modeling constructs to be used.

3.1577
level 1 control break
1. a situation which occurs when the values in the control field elements in the current logical record (and at the level specified) are not the same as the values in the same elements of the immediately previously processed logical record*. ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.8

NOTE The values of the control field elements will be obtained from the current logical record in the initiation routines of the appropriate level. The action of examining the control field should ensure that if there is a change in value at a level above 1, a control break is registered at all levels subordinate to the level at which the change actually occurs.

3.1578
level 1 initiation routine
1. those activities required when level 1 control break occurs to commence processing all records belonging to level 1 record group*. ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.11

3.1579
level 1 record group
1. a set of records whose control field elements are equal for all levels superior to and including level 1. *ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.9

3.1580
level 1 termination routine
1. those activities required when level 1 control break occurs to conclude processing all records belonging to level 1 record group. *ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.1

3.1581
level of abstraction
1. a view of an object at a specific level of detail

3.1582
level of effort (LOE)
1. support activity which does not produce definitive end products, generally characterized by a uniform rate of work performance over a period of time determined by the activities supported

EXAMPLE customer liaison, project cost accounting, project management

3.1583
level of performance
1. the degree to which the needs are satisfied, represented by a specific set of values for the quality characteristics. *ISO/IEC 9126-1:2001, Software engineering — Product quality — Part 1: Quality model.*4.1

3.1584
LF
1. late finish date. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1585
license
1. a 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

3.1586
licensing standard
1. a 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

3.1587
life cycle
1. evolution of a system, product, service, project or other human-made entity from conception through retirement. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.16, *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.11. ***2.*** the system or product evolution initiated by a perceived stakeholder need through the disposal of the products. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.20

3.1588
life cycle cost
1. the total investment in product development, manufacturing, test, distribution, operation, support, training, and disposal. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.21. *Syn:* life-cycle cost

3.1589
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 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.16, *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.12. *Syn:* life-cycle model

3.1590
life cycle processes
1. a set of interrelated activities that result in the development or assessment of software products. *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.18; *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.18. *Syn:* life-cycle processes

NOTE Each activity consists of tasks. The life cycle processes may 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 Software Validation and Verification Plan.

3.1591
lightweight process
1. a process with a single thread of control; a task

3.1592
limited entry table
1. a 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

3.1593
line of code
1. a programming-language statement; a non-comment, nonblank deliverable source statement

3.1594
link
1. to create a load module from two or more independently translated object modules or load modules by resolving cross-references among them ***2.*** a part of a computer program, often a single instruction or address, that passes control and parameters between separate modules of the program ***3.*** to provide a link as in (2). ***4.*** (on-screen documentation) active area that displays either a new topic or a different part of the current topic. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.27

cf. linkage editor

3.1595
linkage editor
1. a 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. *Syn:* linker

cf. linking loader

NOTE may be part of a loader

3.1596
linking loader
1. a 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, linkage editor, relocating loader

3.1597
list
1. a set of data items, each of which has the same data definition. ***2.*** to print or otherwise display a set of data items. ***3.*** a collection class that contains no duplicates and whose members are ordered. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.101

3.1598
list function
1. an online function that displays an overview of entity type occurrences that may or may not have to 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*

3.1599
list processing language
1. a 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 and IPL

3.1600
listing
1. an ordered display or printout of data items, program statements, or other information

3.1601
literal
1. in a source program, an explicit representation of the value of an item. ***2.*** the denotation of a specific instance of a value class. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.102. ***3.*** a number or string that is used by a program directly rather than being embedded in a named constant or variable

3.1602
LITSR
1. level interim test status report. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1603
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

3.1604
load map
1. a computer-generated list that identifies the location or size of all or selected parts of memory-resident code or data

3.1605
load module
1. a 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

3.1606
load-and-go
1. an operating technique in which there are no stops between the loading and execution phases of a computer program

3.1607
loaded origin
1. the 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

3.1608
loader
1. a 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: Types include absolute loader, linking loader, relocating loader.

3.1609
local area network (LAN)
1. a computer network located on a user's premises within a limited geographical area*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.46

NOTE Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary may be subject to some form of regulation.

3.1610
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

3.1611
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

3.1612
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

3.1613
local SAM owner
1. individual at any 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 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 7

3.1614
local variable
1. a variable that can be accessed by only one module or set of nested modules in a computer program

cf. global variable

3.1615
localization
1. creation of a national or specific regional version of a product. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.28

cf. internationalization

NOTE Localization may be carried out separately from the translation process.

3.1616
location facility
1. a set of service primitives that allow a client-side binder object to ask a server-side if it will accept requests carrying invocations to a particular (computational) server object. *ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*3.3.9

NOTE The server-side can confirm or reject the proposal or suggest an alternative server-side that is capable of handling requests.

3.1617
location reference
1. indicator following a heading or subheading in an index, showing to which part of the document the heading or subheading refers. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.27

3.1618
location transparency
1. a distribution transparency which masks the use of information about location in space when identifying and binding to interfaces. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.3

3.1619
lock
1. an exclusive permission to edit a file

3.1620
lockout
1. a 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

3.1621
LOE
1. level of effort. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1622
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) — Fourth Edition*

3.1623
log off
1. to end a session. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.54. *Syn:* log out

3.1624
log on
1. to initiate a session. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.53. *Syn:* log in

3.1625
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

3.1626
logical cohesion
1. a 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

3.1627
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 It is an internal logical file or an external interface file.

3.1628
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 This does not pertain to the physical way in which data is presented on a screen, report, or other media.

3.1629
logical record
1. the set of data which is processed in a single iteration of the main procedure. *ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.3

NOTE It may be part or the whole of a single physical record or of a number of records.

3.1630
logical relationship
1. a dependency between two project schedule activities, or between a project schedule activity and a schedule milestone. The four possible types of logical relationships are: Finish-to-Start; Finish-to-Finish; Start-to-Start; and Start-to-Finish. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* dependency

cf. precedence relationship

3.1631
logical source statement (LSS)
1. software instruction, independent of the physical format (lines of code) in which it appears

cf. physical source statement

3.1632
logical trace
1. an execution trace that records only branch or jump instructions

cf. execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

3.1633
logical transaction
1. the basic functional component of Mk II FPA. *ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10

NOTE The smallest complete unit of information processing that is meaningful to the end user in the business. It is triggered by an event in the real world of interest to the user, or by a request for information. It comprises an input, process and output component. It must be self-contained and leave the application being counted in a consistent state.

3.1634
logical type
1. a 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

3.1635
loop
1. a 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). *Syn:* iterative construct

cf. loop body, loop control, UNTIL, WHILE

3.1636
loop assertion
1. a logical expression specifying one or more conditions that must be met each time a particular point in a program loop is executed. *Syn:* loop invariant

cf. input assertion, output assertion, inductive assertion method

3.1637
loop body
1. the part of a loop that accomplishes the loop's primary purpose

cf. loop control

3.1638
loop control
1. the part of a loop that determines whether to exit from the loop

cf. loop body, leading decision, trailing decision

3.1639
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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.70

NOTE These uses are referred to as input loopback, control loopback, and mechanism loopback, respectively.

3.1640
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

3.1641
loop-control variable
1. a program variable used to determine whether to exit from a loop

3.1642
low level
1. specific; detailed

3.1643
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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.103

3.1644
low-level design
1. the process of design at the individual-routine or, sometimes, class level under the guidance of a more general design. *Syn:* detailed design

3.1645
LS
1. late start date*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1646
LTC
1. level test case*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1647
LTD
1. level test design*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1648
LTL
1. level test log*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1649
LTP
1. level test plan*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

cf. LTPr

3.1650
LTPr
1. level test procedure*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

cf. LTP

3.1651
LTR
1. level test report*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1652
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

3.1653
machine language
1. a language that can be recognized by the processing unit of a computer. *Syn:* first-generation language, machine-oriented language

cf. assembly language, fifth-generation language, fourth-generation language, high-order language, symbolic language

NOTE Such a language usually consists of patterns of 1s and 0s, with no symbolic naming of operations or addresses.

3.1654
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

3.1655
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

3.1656
machine-readable
1. pertaining to data in a form that can be automatically input to a computer. *Syn:* machine readable

EXAMPLE data encoded on a diskette

3.1657
macro
1. in software engineering, a predefined sequence of computer instructions that is inserted into a program, usually during assembly or compilation, at each place that its corresponding macroinstruction appears in the program. *Syn:* macro definition

cf. macroinstruction, macrogenerator, open subroutine

3.1658
macro library
1. a collection of macros available for use by a macrogenerator

cf. system library

3.1659
macroassembler
1. an assembler that includes, or performs the functions of, a macrogenerator

3.1660
macrogenerator
1. a 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. *Syn:* macro-generating program

3.1661
macroinstruction
1. a 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

3.1662
macroprocessor
1. a routine or set of routines provided in some assemblers and compilers to support the definition and use of macros

3.1663
macroprogramming
1. computer programming using macros and macroinstructions

3.1664
magic number
1. literal value that is used by a program directly rather than being embedded in a named constant or variable. *Syn:* magic string

cf. literal

3.1665
main procedure
1. all those activities, subsequent to the general initiation routine and prior to the general termination routine within the complete procedure. *ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups.*2.2

3.1666
main program
1. a software component that is called by the operating system of a computer and that usually calls other software components

cf. routine, subprogram

3.1667
mainframe
1. a computer intended to run in a computer center, with extensive capabilities and resources to which other computers may be connected so that they can share facilities. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.02

3.1668
maintainability
1. the 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.*** the 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.*** the capability of the software product to be modified. *ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.4. ***4.*** the average effort required to locate and fix a software failure. ***5.*** speed and ease with which a program can be corrected or changed. *IEEE Std 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability.*2.3

cf. extendability, flexibility

NOTE Modifications may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications [ISO/IEC 9126-1].

3.1669
maintainability plan
1. a document setting out the specific maintainability practices, resources and sequence of activities relevant to software

NOTE The developer prepares the Maintainability Plan.

3.1670
maintained
1. the ability to modify data through an elementary process, that is, add, change, delete, populate, revise, update, assign, and create*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*.6

3.1671
maintainer
1. individual or organization that performs maintenance activities*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*.4.31. ***2.*** organization that performs maintenance activities*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.18

3.1672
maintenance
1. the process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment ***2.*** the process of retaining a hardware system or component in, or restoring it to, a state in which it can perform its required functions ***3.*** the effort to keep an application performing according to its specifications, generally without changing its functionality (or function point count)*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual. Syn:* software maintenance

cf. adaptive maintenance, corrective maintenance, perfective maintenance

3.1673
maintenance (support) rate
1. the productivity measure for maintaining an application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE It is expressed as the Work Effort by category of maintenance divided by 1000 Application Function Points in a period of time.

3.1674
maintenance branch
1. a branch where most development concerns bug fixes

3.1675
maintenance enhancement
1. a modification to an existing software product to satisfy a new requirement*. ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance*.3.5

NOTE There are two types of software enhancements, adaptive and perfective. A maintenance enhancement is not a software correction.

3.1676
maintenance manual (MM)
1. a software engineering project-deliverable document that enables a system's maintenance personnel (rather than users) to maintain the system

cf. operator manual, user manual

3.1677
maintenance personnel
1. software engineers who maintain software systems

3.1678
maintenance plan
1. a document setting out the specific maintenance practices, resources, and sequence of activities relevant to maintaining a software product

3.1679
maintenance program
1. the organizational structure, responsibilities, procedures, processes, and resources used for implementing the maintenance plan. *Syn:* maintenance infrastructure

3.1680
maintenance project
1. a 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

3.1681
manage project team
1. [Process] the process of tracking team member performance, providing feedback, resolving issues, and managing changes to optimize project performance. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1682
manage stakeholder expectations
1. [Process] the process of communicating and working with stakeholders to meet their needs and addressing issues as they occur. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1683
managed network
1. a network or set of networks established and controlled by one or more organizations to meet specific organizational or business needs. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle*.3.1.6

3.1684
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

3.1685
managed web site
1. a site created and maintained based on organizational guidelines. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle*.3.1.7

3.1686
management
1. system of controls and processes required to achieve the strategic objectives set by the organization's governing body. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.9

NOTE Management is subject to the policy guidance and monitoring set through corporate governance.

3.1687
management process
1. the activities that are undertaken in order to ensure that the software engineering processes are performed in a manner consistent with the organization's policies, goals, and standards*. ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)*.8

3.1688
management review
1. a 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 Std 1028-2008 IEEE Standard for Software Reviews and Audits*.3.4

3.1689
mandatory
1. a syntax keyword used to specify a total mapping*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.104

cf. optional, total

3.1690
mandatory category
1. category that is essential to establish a common definition and to provide common terminology and concepts for communication among projects, business environments, and personnel*. IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies*.3.5

3.1691
mandatory nonidentifying relationship
1. a nonidentifying relationship in which an instance of the child entity must be related to an instance of the parent entity*. IEEE Std 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]

3.1692
mandatory subgroup
1. one of the two types of subgroups for record element types (RETs)*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE Mandatory subgroups mean the user must use one of the subgroups during an elementary process that creates an instance of the data.

3.1693
manufacture
1. in software engineering, the process of copying software to disks, chips, or other devices for distribution to customers or users

3.1694
manufacturer
1. the organization that develops the software package*. ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages*.3.2.4

3.1695
manufacturing phase
1. the 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

3.1696
many-sorted algebra
1. a 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

3.1697
many-to-many relationship
1. a relationship between two state classes (not necessarily distinct) in which each instance of one class may be associated with any number of instances of a second class (possibly none), and each instance of the second class may be related to any number of instances of the first class (possibly none). *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.106

3.1698
map program
1. a software tool, often part of a compiler or assembler, that generates a load map

3.1699
mapping
1. an assigned correspondence between two things that is represented as a set of ordered pairs. *IEEE Std 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). *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*Annex E

cf. instance, invocation, iteration, software life cycle model (SLCM)

3.1700
mapping completeness
1. a designation of whether a mapping is complete (totally mapped) or incomplete (partial). *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.108

cf. partial, total

3.1701
marketing organization
1. the organization that markets the software package. *ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.5

3.1702
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

3.1703
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

3.1704
mark-up
1. document with comments written on it indicating changes that need to be made. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.28. ***2.*** process of producing such a document as in (1). ISO/IEC 15910:1999, *Information technology — Software user documentation process*. 4.28

3.1705
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.

3.1706
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

3.1707
master schedule
1. [Tool] 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) — Fourth Edition*

cf. milestone schedule

3.1708
material
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) — Fourth Edition. Syn:* materiel

3.1709
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) — Fourth Edition*

3.1710
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

3.1711
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

3.1712
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.*4.10

3.1713
mean time between failures (MTBF)
1. the expected or observed time between consecutive failures in a system or component

cf. up time

3.1714
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

3.1715
meaning (of a responsibility)
1. a statement of what the responsibility means. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.109

NOTE 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 should be complete enough to let a requester decide whether to make the request, but it should stop 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 may be more formally refined into a statement of pre-conditions and post-conditions using the specification language.

3.1716
measurable concept
1. abstract relationship between attributes of entities and information needs. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process*.3.14

3.1717
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 The specification of a measurand may require statements about quantities such as time, temperature and pressure. [International vocabulary of basic and general terms in metrology, 1993, definition 2.6].

3.1718
measure
1. variable to which a value is assigned as the result of measurement. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process*.3.15*; ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*.4.32. ***2.*** make a measurement. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process*.3.16; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE*.4.33. ***3.*** a way to ascertain or appraise value by comparing it to a norm. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*.2.6. ***4.*** to apply a metric. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*.2.6. ***5.*** a number that assigns relative value. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***6.*** to ascertain or appraise by comparing to a standard. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***7.*** the number or category assigned to an attribute of an entity by making a measurement. *ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview*.4.18. ***8.*** the 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 Std 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*.2.4. ***9.*** the act or process of measuring. *IEEE Std 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability*.2.4

3.1719
measure of effectiveness (MOE)
1. the metrics by which an acquirer will measure satisfaction with products produced by the technical effort. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.23

3.1720
measure of performance (MOP)
1. an engineering performance measure that provides design requirements that are necessary to satisfy an MOE. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.24

NOTE There are generally several measures of performance for each MOE

3.1721
measurement
1. act or process of assigning a number or category to an entity to describe an attribute of that entity. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.7. ***2.*** the assignment of numbers to objects in a systematic way to represent properties of the object. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*9.4.2. ***3.*** the use of a metric to assign a value (which may be a number or category) from a scale to an attribute of an entity. *ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.19. ***4.*** set of operations having the object of determining a value of a measure. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.34. ***5.*** the assignment of values and labels to aspects of software engineering (products, processes, and resources) and the models that are derived from them, whether these models are developed using statistical, expert knowledge or other techniques. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*8. ***6.*** assigning relative value. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***7.*** a figure, extent, or amount obtained by measuring. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.7

3.1722
measurement analyst
1. individual or organization that is responsible for the planning, performance, evaluation, and improvement of measurement. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.18

3.1723
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 15939:2007, Systems and software engineering — Measurement process.*3.19

3.1724
measurement function
1. algorithm or calculation performed to combine two or more base measures. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.20, *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.35

3.1725
measurement librarian
1. individual or organization that is responsible for managing the measurement data store(s). *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.21

3.1726
measurement method
1. logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.22; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.36

NOTE The type of measurement method depends on the nature of the operations used to quantify an attribute. Two types may be distinguished: subjective - quantification involving human judgment; objective - quantification based on numerical rules.

3.1727
measurement procedure
1. set of operations, described specifically, used in the performance of a particular measurement according to a given method. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.23; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.37

3.1728
measurement process
1. process for establishing, planning, performing and evaluating software measurement within an overall project or organizational measurement structure. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.24; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.38

3.1729
measurement process owner
1. individual or organization responsible for the measurement process. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.25

3.1730
measurement sponsor
1. individual or organization that authorizes and supports the establishment of the measurement process. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.26

3.1731
measurement standard
1. a standard that describes the characteristics of evaluating a process or product

3.1732
measurement user
1. individual or organization that uses the information products. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.27

3.1733
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

3.1734
mechanicals
1. printing, binding, production and layout details for paper-based documentation. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.29

3.1735
mechanism
1. in an IDEF0 model, the means used by a function to transform input into output. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.71

3.1736
mechanism arrow
1. an arrow or arrow segment that expresses IDEF0 mechanism. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.72

NOTE That is, an object type set whose instances are used by a function to transform input into output. The arrowhead of a mechanism arrow is attached to the bottom side of a box.

3.1737
mechanism loopback
1. loopback of output from one function to be mechanism for another function in the same diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.73

3.1738
medium
1. a channel of communication or information, plural is media. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

EXAMPLE a report issued on paper or in microfiche

3.1739
memory
1. the addressable storage space in a processing unit and all other internal storage that is used to execute instructions. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.11

3.1740
memory capacity
1. the maximum number of items that can be held in a given computer memory; usually measured in words or bytes

cf. channel capacity, storage capacity

3.1741
memory compaction
1. a 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.*** a storage allocation technique in which contiguous blocks of non-allocated storage are combined to form single blocks. *Syn:* garbage collection

3.1742
memory dump
1. a 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

3.1743
memory map
1. a diagram that shows where programs and data are stored in a computer's memory

3.1744
menu
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.06. ***3.*** (on-screen documentation) list of topics from which the user may choose. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.29

3.1745
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

3.1746
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*

3.1747
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

3.1748
merge
1. to combine different changes to the same file

NOTE Many systems follow the optimistic strategy of combining all lines that do not conflict.

3.1749
merge from current
1. to merge changes from the current branch into the stable branch(es)

NOTE To avoid disruptive changes in a stable branch, code changes are typically first introduced into the current (development) branch, tested, and then merged back.

3.1750
message
1. a communication sent from one object to another. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.110

cf. request

NOTE Message encompasses requests to meet responsibilities as well as simple informative communications.

3.1751
meta-
1. a prefix to a concept to imply definition information about the concept. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.4.2

NOTE Specifically, used to designate the location of an object in the three model layers.

3.1752
meta-attribute
1. a definition of a characteristic of a meta-entity or meta-relationship. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.4.2

NOTE Instances of a meta-attribute occur in a model as data values.

3.1753
meta-entity
1. a definition of a type of data object that occurs in CDIF models. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview*.4.2

NOTE Specifically, a meta-entity represents a set of zero or more meta-attributes, stored together to represent a thing, event or concept that has instances in a model.

3.1754
metalanguage
1. a language used to specify some or all aspects of a language

cf. stratified language, unstratified language

EXAMPLE Backus-Naur form

3.1755
meta-meta-attribute
1. a definition of a characteristic of a meta-meta-entity or meta-meta-relationship Instances of a meta-meta-attribute occur in a metamodel as meta-data values. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

3.1756
meta-meta-entity
1. a definition of the behavior and structure of meta-entities, meta-relationships, meta-attributes, or subject areas. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE i.e., a definition of the meta-object definitions used to describe information in models

3.1757
meta-meta-relationship
1. a definition of a type of data object that occurs in CDIF metamodels. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE Specifically, a meta-meta-relationship represents the definition of a relationship between instances of meta-meta-entities.

3.1758
metamodel
1. a logical information model that specifies the modeling elements used within another (or the same) modeling notation. *IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections — Classification and Description.*3.8. ***2.*** a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.111. ***3.*** a 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 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.4. *Syn:* meta-model

3.1759
meta-object
1. a generic term for meta-entities, meta-relationships and meta-attributes. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

3.1760
meta-object facility
1. a 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

3.1761
meta-relationship
1. a definition of a type of data object that occurs in CDIF models. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE Specifically, a meta-relationship represents the definition of a relationship between meta-entities that has instances in a model. A meta-relationship may also define a set of zero or more meta-attributes, stored together to represent characteristics of a relationship between meta-entities.

3.1762
method
1. an implementation of an operation*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.21. ***2.*** a statement of how property values are combined to yield a result*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.112

3.1763
method engineer
1. person who designs, builds, extends and maintains methodologies*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.1

NOTE Method engineers create methodologies from metamodels via generation.

3.1764
method standard
1. a standard that describes the characteristics of the orderly process or procedure used in the engineering of a product or performing a service

3.1765
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) — Fourth 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 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.2

NOTE A methodology specifies the process to be executed, usually as a set of related activities, tasks and/or techniques, together with the work products that must be manipulated (created, used or changed) at each moment and by whom, possibly including models, documents and other inputs and outputs. In turn, specifying the models that must be dealt with implies defining the basic building blocks that should be used to construct.

3.1766
methodology element
1. simple component of a methodology*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*3.6

NOTE Usually, methodology elements include the specification of what tasks, activities, techniques, models, documents, languages and/or notations can or must be used when applying the methodology. Methodology elements are related to each other, comprising a network of abstract concepts. Typical methodology elements are Capture Requirements, Write Code for Methods (kinds of tasks), Requirements Engineering, High-Level Modeling (kinds of activities), Pseudo-code, Dependency Graphs (notations), Class, Attribute (kinds of model building blocks), Class Model, Class Diagram, Requirements Specification (kind of work products).

3.1767
metric
1. a combination of two or more measures or attributes*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** a quantitative measure of the degree to which a system, component, or process possesses a given attribute. ***3.*** the defined measurement method and the measurement scale*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.20

cf. software quality metric

3.1768
metric validation
1. the act or process of ensuring that a metric reliably predicts or assesses a quality factor*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.11

3.1769
metric value
1. a metric output or an element that is from the range of a metric*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.12

3.1770
metrics framework
1. a decision aid used for organizing, selecting, communicating, and evaluating the required quality attributes for a software system*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.9. ***2.*** a hierarchical breakdown of quality factors, quality subfactors, and metrics for a software system*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.9

3.1771
metrics sample
1. a set of metric values that is drawn from the metrics database and used in metrics validation*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.1

3.1772
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.26

cf. KOPS, MIPS

NOTE: a measure of computer processing speed

3.1773
micro code assembler
1. a computer program that translates microprograms from symbolic form to binary form

3.1774
microarchitecture
1. the microword definition, data flow, timing constraints, and precedence constraints that characterize a given microprogrammed computer

3.1775
microcode
1. a collection of microinstructions, comprising part of or all of microprograms*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.19. *Syn:* micro code

3.1776
microcomputer
1. a digital computer whose processing unit consists of one or more microprocessors, and includes storage and input-output facilities*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.19

3.1777
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 Types include diagonal microinstruction, horizontal microinstruction, vertical microinstruction.

3.1778
microoperation
1. in microprogramming, one of the basic operations needed to carry out a machine language instruction

cf. microinstruction

3.1779
microprocessor
1. a processor whose elements have been miniaturized into one or a few integrated circuits. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.09

3.1780
microprogram
1. a sequence of instructions, called microinstructions, specifying the basic operations needed to carry out a machine language instruction. *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*

3.1781
microprogrammable computer
1. a microprogrammed computer in which microprograms can be created or altered by the user

3.1782
microprogrammed computer
1. a computer in which machine language instructions are implemented by microprograms rather than by hard-wired logic

cf. microarchitecture, microprogrammable computer

NOTE A microprogrammed computer may or may not be a microcomputer; the concepts are not related despite the similarity of the terms.

3.1783
microprogramming
1. the 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, microprogram

NOTE This method is an alternative to hard-wiring the control signals necessary to carry out each machine language instruction. Techniques include bit steering, compaction, residual control, single-level encoding, two-level encoding.

3.1784
microword
1. an addressable element in the control store of a microprogrammed computer

3.1785
migrated attribute
1. a foreign key attribute of a child entity. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.113

NOTE [key style]

3.1786
migration
1. moving a cluster to a different capsule. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.27

3.1787
migration transparency
1. a distribution transparency which masks from an object, the ability of a system to change the location of that object. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.4

NOTE Migration is often used to achieve load balancing and reduce latency.

3.1788
milestone
1. a significant point or event in the project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** a scheduled event used to measure progress*. IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.3

EXAMPLE Major milestones for software projects may include an acquirer or managerial sign-off, baselining of a specification, completion of system integration, and product delivery. Minor milestones might include baselining of a software module or completion of a chapter of the user manual

3.1789
milestone schedule
1. [Tool] a summary-level schedule that identifies the major schedule milestones*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. master schedule

3.1790
MIM
1. Management Information Model*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1791
minicomputer
1. a digital computer that is functionally intermediate between a microcomputer and a mainframe*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.23

NOTE Servers and network devices have generally replaced minicomputers.

3.1792
minimum delay programming
1. a programming technique in which storage locations for computer instructions and data are chosen so that access time is minimized

3.1793
minimum tasks
1. those tasks required for the integrity level assigned to the software to be tested*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.19

3.1794
MIPS
1. million instructions per second

cf. KOPS, MFLOPS

NOTE a measure of computer processing speed

3.1795
mirror site
1. a duplicate copy of a master site maintained on a different host typically to provide redundancy, higher performance, or local access*. IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.8

3.1796
MIS
1. Management Information Systems*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*4

3.1797
mistake
1. a human action that produces an incorrect result

NOTE The fault tolerance discipline distinguishes between a human action (a mistake), its manifestation (a hardware or software fault), the result of the fault (a failure), and the amount by which the result is incorrect (the error).

3.1798
mitigating function
1. a function that, if provided successfully, will prevent an initiating event from becoming a specified threat. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.11

3.1799
mixed entry table
1. a 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

3.1800
mixed mode
1. pertaining to an expression that contains two or more different data types. *Syn:* mixed type

EXAMPLE Y = X + N, where X and Y are floating point variables and N is an integer variable

3.1801
MMC(S)
1. Multimedia Conferencing (System). *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1802
MMI
1. man-machine interface

cf. user interface

3.1803
mobility schema
1. a specification putting constraints on the mobility of an object. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*16.4.1.1

3.1804
mock object
1. temporary dummy objects created to aid testing until the real objects become available

3.1805
mock-up
1. a 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 It can be retained for verification or training, and as a record.

3.1806
mode
1. a set of related features or functional capabilities of a product. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.1. ***2.*** a 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

3.1807
model
1. a representation of a real world process, device, or concept. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.11. ***2.*** a representation of something that suppresses certain aspects of the modeled subject. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.115. ***3.*** an interpretation of a theory for which all the axioms of the theory are true. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.115. ***4.*** a 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.*** a semantically closed abstraction of a system or a complete description of a system from a particular perspective

3.1808
model glossary
1. the collection of the names and definitions of all defined concepts that appear within the views of a model. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.116. ***2.*** the diagrams that correspond to the nodes of the hierarchical graph structure of an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.74

3.1809
model hierarchy
1. the diagrams that correspond to the nodes of the hierarchical graph structure of an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.74

3.1810
model layers
1. the different layers of definition (or abstraction) used in defining the CDIF family of standards. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE The four model layers in CDIF are user data, model, metamodel, meta-metamodel. Any given model layer provides an accurate and complete definition of all the instances that may 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.

3.1811
model name
1. a unique, descriptive name that distinguishes one IDEF0 model from other IDEF0 models with which it may be associated. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.75

NOTE An IDEF0 model's model name and model name abbreviation are placed in the A-0 context diagram along with the model's purpose statement and viewpoint statement.

3.1812
model name abbreviation
1. a unique short form of a model name that is used to construct diagram references. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.76

3.1813
model note
1. a textual and/or graphical component of a diagram that records a fact not otherwise depicted by a diagram's boxes and arrows. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.77

3.1814
model note number
1. an integer number, placed inside a small square, that unambiguously identifies a model note in a specific diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.78

3.1815
model page
1. a logical component of an IDEF0 model that can be presented on a single sheet of paper. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.79

NOTE Model pages include diagram, text, FEO, and glossary pages.

3.1816
modeling tool
1. a 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

3.1817
modifiability
1. the ease with which a system can be changed without introducing defects

cf. maintainability

3.1818
modifiable
1. structured and has a style such that changes can be made completely, consistently, and correctly while retaining the structure. *IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*H.3 e)

3.1819
modification request (MR)
1. proposed changes to a product that is being maintained. *ISO/IEC 14764:2006 (IEEE Std 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. *Syn:* change request

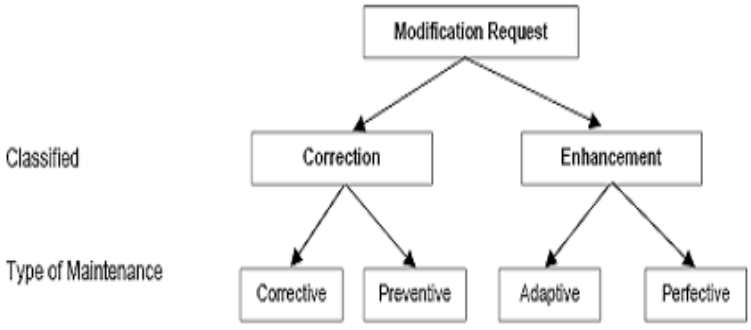


Figure 16 — Modification request

NOTE The MR may later be classified as a correction or enhancement and identified as corrective, preventive, adaptive, or perfective maintenance. Trouble/problem-reporting documents include incident and trouble reports; configuration change control documents include software change requests (SCRs).

3.1820
modified source statement
1. source statement that has been changed from the original source

3.1821
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 Std 1062, 1998 Edition (R2002) IEEE Recommended Practice for Software Acquisition (includes IEEE Std 1062a)*.3.6

3.1822
MODL
1. Meta-Object Definition Language. *ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*4

3.1823
modular
1. composed of discrete parts

cf. modular decomposition, modular programming

3.1824
modular decomposition
1. the process of breaking a system into components to facilitate design and development; an element of modular programming. *Syn:* modularization

cf. cohesion, coupling, demodularization, factoring, functional decomposition, hierarchical decomposition, packaging

3.1825
modular programming
1. a 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

3.1826
modularity
1. the 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 ***2.*** software attributes that provide a structure of highly independent components. ***3.*** the extent to which a routine or module is like a black box

cf. cohesion, coupling

3.1827
module
1. a program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading. ***2.*** a logically separable part of a program. ***3.*** a set of source code files under version control that can be manipulated together as one. ***4.*** a collection of both data and the routines that act on it

NOTE The terms 'module', 'component,' and 'unit' are often used interchangeably or defined to be subelements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized.

3.1828
module data
1. data that can be accessed by any routine within the module in which it is declared but not by routines in other modules. *Syn:* instance data, class data

3.1829
MOF
1. Meta-Object Facility*. ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*4

3.1830
monadic selective construct
1. an 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

3.1831
monitor
1. a 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) — Fourth Edition. Syn:* execution monitor

cf. hardware monitor, software monitor

3.1832
monitor and control project work
1. [Process] 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) — Fourth Edition*

3.1833
monitor and control risks
1. [Process] the process of implementing risk response plans, tracking identified risks, monitoring residual risks, identifying new risks, and evaluating risk process throughout the project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1834
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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.19

3.1835
monitoring and controlling processes
1. [Process Group] 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) — Fourth Edition*

3.1836
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) — Fourth Edition*

3.1837
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) — Fourth Edition*

3.1838
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 may differ from that of the source

cf. copy

EXAMPLE to move data from one file to another

3.1839
MTBF
1. mean time between failures

3.1840
MTP
1. master test plan. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.2

3.1841
MTR
1. master test report. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.2

3.1842
MTTR
1. mean time to repair

3.1843
multiaddress instruction
1. a computer instruction that contains more than one address field. *Syn:* multiple-address instruction

cf. one-address instruction

3.1844
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

3.1845
multiple inclusive selective construct
1. a 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

3.1846
multiple inheritance
1. the ability of a subclass to inherit responsibilities from more than one superclass. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.119. ***2.*** the situation when the subtype inherits all of the meta-attributes and meta-relationships of all of its supertypes (and their supertypes, etc). *ISO/IEC 15474-2:2002, Information technology — CDIF framework — Part 2: Modelling and extensibility*.6.2.6

3.1847
multiple readers and writers
1. an algorithm that lets multiple readers access a shared data repository concurrently; however, writers must have mutually exclusive access to update the repository

3.1848
multiple sites GSC
1. one of the 14 general system characteristics describing the degree to which the application has been developed for multiple locations and user organizations. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1849
multiple-hit decision table
1. a 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

3.1850
multiplicity
1. a 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

3.1851
multiprocessing
1. a 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

3.1852
multiprogramming
1. a 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

3.1853
multiset
1. a 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

3.1854
multiset cardinality
1. the 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. *Syn:* cardinality of a multiset

3.1855
multitasking
1. a mode of operation in which two or more tasks are executed in an interleaved manner

cf. multiprocessing, multiprogramming, time sharing

3.1856
multi-valued
1. a mapping that is not a function. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.117

cf. function

3.1857
multi-valued property
1. a property with a multi-valued mapping. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.118

cf. single-valued property

3.1858
mutable class
1. a class for which the set of instances is not fixed; its instances come and go over time. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.120

cf. immutable class, state class

3.1859
mutation testing
1. a 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

3.1860
mutual exclusion
1. giving access to shared data only to one task at a time

NOTE can be enforced by means of binary semaphores or by using monitors

3.1861
mutually exclusive clustering
1. a 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

3.1862
N 2 diagram
1. a system engineering or software engineering tool for tabulating, defining, analyzing, and describing functional interfaces and interactions among system components. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document*.3.1

NOTE The N 2 diagram is a matrix structure that graphically displays the bidirectional interrelationships between functions and components in a given system or structure.

3.1863
n-address instruction
1. a computer instruction that contains n address fields, where n may be any non-negative integer

cf. one-address instruction, two-address instruction, n-plus-one address instruction

3.1864
name
1. a word or phrase that designates some model construct. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.121

NOTE Such as a class, responsibility, subject domain, etc.

3.1865
named constant
1. an identifier that refers to a numeric or string value that does not change during program execution

3.1866
named constraint
1. a constraint that is specific to a particular model, rather than being inherent in some modeling construct. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.122

NOTE Such as a cardinality constraint. A named constraint is explicitly named, its meaning is stated in natural language, and its realization is written in the specification language.

3.1867
nano code
1. a collection of nanoinstructions

3.1868
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

3.1869
nanostore
1. in a two-level implementation of microprogramming, a secondary control store in which nanoinstructions reside

3.1870
N-ary relationship
1. a relationship with arity (degree) n > 2. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE A relationship that has more than two participating entities. (Note that a single entity may participate several times in a single relationship.)

3.1871
natural language
1. a language whose rules are based on usage rather than being pre-established prior to the language's use. ***2.*** a language whose rules are based on current usage without being specifically prescribed. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.08

cf. formal language

EXAMPLE German and English

3.1872
navigation
1. means by which a user moves from one part of a software application to another. ***2.*** act of accessing documentation and viewing different topics. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.30

3.1873
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 dialog faster

3.1874
NDI
1. non-developmental item. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.1875
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) — Fourth Edition*

3.1876
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

3.1877
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

3.1878
net
1. a 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

3.1879
net graph
1. a 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

3.1880
network
1. an arrangement of nodes and interconnecting branches. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.44

3.1881
network chart
1. a directed graph used for describing and scheduling events, activities, and their relationships in project control. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.05

3.1882
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) — Fourth Edition*

3.1883
network open end
1. schedule activity without any predecessor activities or successor activities, causing a break in a schedule network path

NOTE Network open ends are usually caused by missing logical relationships.

3.1884
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) — Fourth Edition*

3.1885
network planning
1. a technique that uses network charts for planning, scheduling. and controlling a project. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.06

3.1886
networking
1. developing relationships with persons who may be able to assist in the achievement of objectives and responsibilities

3.1887
new source statements
1. the sum of the added and modified source statements

3.1888
n-level address
1. an 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

3.1889
node
1. in a diagram, a point, circle, or other geometric figure used to represent a state, event, or other item of interest. ***2.*** a 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.7. ***3.*** a modeled function located within the hierarchical graph structure of an IDEF0 model by its designated node number. *IEEE Std 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) — Fourth Edition.* ***5.*** a 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); a parallel computer the control of a single operating system

NOTE A node may have internal structure which is not of concern in an engineering specification.

3.1890
node index
1. a text listing, often indented, of the nodes in an IDEF0 model, shown in outline order. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.81

NOTE Same meaning and node content as a node tree

3.1891
node letter
1. the letter that is the first character of a node number. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.82

3.1892
node number
1. an expression that unambiguously identifies a function's position in a model hierarchy. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.83

NOTE A node number is constructed by concatenating a node letter, the diagram number of the diagram that contains the box that represents the function, and the box number of that box.

3.1893
node tree
1. a graphical listing of the nodes of an IDEF0 model, showing parent-child relationships as a graphical tree. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.84

NOTE same meaning and node content as a node index

3.1894
nomenclature standard
1. a standard that describes the characteristics of a system or set of names, or designations, or symbols

3.1895
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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.20. *Syn:* nondeliverable item

3.1896
nondelivered source statement
1. source statement that is developed in support of the final product, but not delivered to the customer. *Syn:* non-delivered source statement

3.1897
nondestructive read
1. a read operation that does not erase the data in the accessed location. *Syn:* non-destructive read

cf. destructive read

3.1898
nondeveloped source statement
1. existing source statement that is reused or deleted. *Syn:* non-developed source statement

3.1899
nondevelopmental
1. developed prior to its current use in an acquisition or development process

NOTE Such an item may require minor modifications to meet the requirements of its current intended use.

3.1900
nonfunctional requirement
1. a software requirement that describes not what the software will do but how the software will do it. *Syn:* design constraints, non-functional requirement

cf. functional requirement

EXAMPLE software performance requirements, software external interface requirements, software design constraints, and software quality attributes. Nonfunctional requirements are sometimes difficult to test, so they are usually evaluated subjectively

3.1901
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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.123. *Syn:* non-identifying relationship

cf. identifying relationship, mandatory nonidentifying relationship, optional nonidentifying relationship [key style]

3.1902
nonintrinsic relationship
1. a relationship that is partial, is multi-valued, or may change*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.125. *Syn:* non-intrinsic relationship

cf. intrinsic relationship

3.1903
nonkey attribute
1. an attribute that is not the primary or a part of a composite primary key of an entity*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.125. *Syn:* non-key attribute

NOTE [key style]

3.1904
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 Non-primary entities have only very few attributes, e.g. code, description.

3.1905
nonprocedural language
1. a 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

3.1906
nonprocedural programming language
1. a computer programming language used to express the parameters of a problem rather than the steps in a solution*. IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2. *Syn:* non-procedural programming language

cf. procedural programming language

EXAMPLE report writer or sort specification languages

3.1907
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

3.1908
non-terminal symbol
1. a 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

3.1909
non-time-critical computationally intensive task
1. a low-priority compute-bound task that consumes spare CPU cycles

3.1910
no-op
1. abbreviation for no-operation

3.1911
no-operation
1. a computer operation whose execution has no effect except to advance the instruction counter to the next instruction. *Syn:* do-nothing operation

NOTE used to reserve space in a program or, if executed repeatedly, to wait for a given event; often abbreviated no-op

3.1912
NOR
1. in configuration management, a notice of revision

3.1913
normalization
1. the process by which any data structure can be transformed by a database designer into a set of normalized relations that have no repeating groups*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1914
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

3.1915
notation standard
1. a standard that describes the characteristics of formal interfaces within a profession

3.1916
note
1. helpful hint or other information that may assist the user by emphasizing or supplementing important points of the main text*. IEEE Std 1063-2001 (R2007) IEEE Standard for Software User Documentation.*2.7. ***2.*** a body of free text that describes some general comment or specific constraint about a portion of a model*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.126

cf. caution, warning

3.1917
notebook computer
1. battery-powered portable computer small and light enough to be operated anywhere*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.22. *Syn:* laptop computer

3.1918
notice of revision (NOR)
1. a 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

3.1919
n-plus-one address instruction
1. a 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

3.1920
nucleus
1. an engineering object which coordinates processing, storage and communications functions for use by other engineering objects within the node to which it belongs. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.6

EXAMPLE an operating system (kernel)

3.1921
numeric
1. pertaining to data that consists of numerals as well as functional units that use the data. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.03

3.1922
object
1. an encapsulation of data and services that manipulate that data. ***2.*** a program constant or variable. ***3.*** a specific entity that exists in a program at runtime in object-oriented programming. ***4.*** a member of an object set and an instance of an object type. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*2.1.85. ***5.*** pertaining to the outcome of an assembly or compilation process

cf. object code, object module, object program

NOTE An object represents something in the observable world that may be distinguished from other instances of its object type and may be uniquely identified.

3.1923 .
object adapter
1. the ORB component which provides object reference, activation, and state related services to an object implementation. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.22

NOTE There may be different adapters provided for different kinds of implementations.

3.1924
object code
1. computer instructions and data definitions in a form output by an assembler or compiler

cf. source code

NOTE An object program is made up of object code.

3.1925
object identifier
1. some concrete representation for the identity of an object (instance). *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.128. *Syn:* oid

NOTE The object identifier (oid) is used to show examples of instances with identity, to formalize the notion of identity, and to support the notion in programming languages or database systems.

3.1926
Object Management Group (OMG)
1. an international standards organization that owns and maintains CORBA and UML standards

3.1927
object model
1. an integrated abstraction that treats all activities as performed by collaborating objects and encompassing both the data and the operations that can be performed against that data. *IEEE/EIA 12207.2-1997 IEEE/EIA*

*Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Implementation considerations.*3.1.129

NOTE An object model captures both the meanings of the knowledge and actions of objects behind the abstraction of responsibility.

3.1928
object module
1. a computer program or subprogram that is the output of an assembler or compiler

cf. load module, object program

3.1929
object of interest (-type)
1. as identified from the point of view of the functional user requirements, any physical thing, as well as any conceptual object or part of a conceptual object in the world of the user about which the software is required to process and/or store data*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.17

NOTE An object of interest is a synonym of 'entity-type' on an entity-relationship diagram, and has the same meaning as the subject of a relation in Third Normal Form.

3.1930
object program
1. a computer program that is the output of an assembler or compiler. *Syn:* target program

cf. source program object module

3.1931
object reference
1. a value that unambiguously identifies an object*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.24

NOTE Object references are never reused to identify another object.

3.1932
object set
1. a subset of instantiations from the set of all possible instantiations of all object types within an object type set*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.86

NOTE An object set is a subset of the union of the members of an object type set; the set of object sets includes the empty set and the set of the union of the members of the object type set itself. An object set is modeled by an arrow segment.

3.1933
object type
1. the set of all possible instantiations of a singular concept, either physical or data, within an IDEF0 model*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.87

NOTE An IDEF0 object type is generally analogous to an IDEF1X entity or an IDEF1 entity class.

3.1934
object type set
1. a named set of one or more object types*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.88

NOTE An object type set may include object types that are themselves grouped as object type sets. An object type set is designated by an arrow label.

3.1935
objective
1. something toward which work is to be directed, a strategic position to be attained, or a purpose to be achieved, a result to be obtained, a product to be produced, or a service to be performed. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** practical advantage or intended effect, expressed as preferences about future states. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.2.1. *Syn:* purpose

NOTE Some objectives are ongoing, some are achieved once met. The enterprise language systematically uses the term objective, (rather than purpose) and emphasizes the need of expressing objective in measurable terms.

3.1936
objective evidence
1. data supporting the existence or verity of something. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.24

NOTE Objective evidence may be obtained through observation, measurement, test, or other means. [ISO 9000:2000]

3.1937
object-oriented design
1. a 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

3.1938
object-oriented language
1. a programming language that allows the user to express a program in terms of objects and messages between those objects

EXAMPLE Smalltalk and LOGO

3.1939
objref
1. abbreviation for object reference. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.25

3.1940
OBS
1. organizational breakdown structure. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.1941
observation
1. instance of applying a measurement procedure to produce a value for a base measure. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.29; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.39

3.1942
observation period
1. the 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

3.1943
occupational title standard
1. a standard that describes the characteristics of the general areas of work or profession

3.1944
OCL
1. Object Constraint Language*. ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*4

3.1945
octet
1. a byte that consists of eight bits*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.10. *Syn:* 8-bit byte

3.1946
ODP
1. Open Distributed Processing*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1947
ODP function
1. a function required to support Open Distributed Processing*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.3.1

3.1948
ODP IDL
1. Open Distributed Processing Interface Definition Language*. ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*4

3.1949
ODP standards
1. this reference model and those standards that comply with it, directly or indirectly*. ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.2

3.1950
ODP system
1. a system which conforms to the requirements of ODP standards*. ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.4

3.1951
ODP-RM
1. Open Distributed Processing: Reference Model*. ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*4. *Syn:* RM-ODP

3.1952
office automation (OA)
1. the integration of office activities by means of an information processing system*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.09

NOTE This term includes in particular the processing and communication of text, images, and voice.

3.1953
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.42

cf. online (2)

3.1954
offset
1. the difference between the loaded origin and the assembled origin of a computer program. ***2.*** a number that must be added to a relative address to determine the address of the storage location to be accessed. *Syn:* relocation factor

3.1955
off-the-shelf
1. already developed and available*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.21

3.1956
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

3.1957
OMG
1. Object Management Group*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1958
OMT
1. Object Modeling Technique*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.1959
one-address instruction
1. a computer instruction that contains one address field. *Syn:* single-address instruction, single-operand instruction

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

3.1960
one-ahead addressing
1. a 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

3.1961
one-plus-one address instruction
1. a 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

3.1962
one-to-many relationship
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.131

3.1963
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.41

cf. batch, conversational, real time

EXAMPLE an airline reservation system

3.1964
online data entry GSC
1. one of the 14 general system characteristics describing the degree to which data is entered through interactive transactions*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1965
online documentation
1. information accessed by the user through the use of software, but that may not be sensitive to context*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.31

cf. help text

3.1966
online update GSC
1. one of the 14 general system characteristics describing the degree to which internal logical files are updated online*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1967
on-screen documentation
1. documentation that is intended to be read on the screen by the user while using the software*. ISO/IEC 26514, 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

3.1968
ontology
1. a logical structure of the terms used to describe a domain of knowledge, including both the definitions of the applicable terms and their relationships*. IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description.*3.9

3.1969
OOD
1. object-oriented design

3.1970
OPA
1. organizational process asset*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process*

3.1971
open distributed processing
1. distributed processing designed to conform to ODP standards. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.3

3.1972
open subroutine
1. a subroutine that is copied into a computer program at each place that it is called. *Syn:* direct insert subroutine

cf. closed subroutine, inline code, macro

3.1973
operable
1. state of being able to perform the intended function

3.1974
operand
1. a 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

3.1975
operating environment (software)
1. the set of software operating concurrently on a specified computer system. *ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.18

3.1976
operating system
1. a 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

3.1977
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.*** the process of running a computer system in its intended environment to perform its intended functions. ***4.*** an interaction between a client object and a server object which is either an interrogation or an announcement. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.2. ***5.*** a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.132. ***6.*** an action needed to perform an activity. *ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.5. ***7.*** arithmetic or logical operation performed in an algorithmic and manipulation BFC. *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.7

EXAMPLE in the expression A = B + 3, the process of adding B to 3 to obtain A

3.1978
operation and maintenance phase
1. the 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

3.1979
operation code
1. a character or set of characters that specifies a computer operation. *Syn:* op code

EXAMPLE the code BNZ to designate the operation 'branch if not zero'

3.1980
operation exception
1. an exception that occurs when a program encounters an invalid operation code

cf. addressing exception, data exception, overflow exception, protection exception, underflow exception

3.1981
operation field
1. the field of a computer instruction that specifies the operation to be performed. *Syn:* function field, operation part

cf. address field

3.1982
operation interface
1. an interface in which all the interactions are operations*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.7

3.1983
operation interface signature
1. an interface signature for an operation interface*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.12

NOTE An operation interface signature comprises a set of announcements and interrogation signatures as appropriate, one for each operation type in the interface, together with an indication of causality (client or server, but not both) for the interface as a whole, with respect to the object which instantiates the template.

3.1984
operational
1. pertaining to a system or component that is ready for use in its intended environment*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.17. ***2.*** pertaining to a system or component that is installed in its intended environment*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.20. ***3.*** pertaining to the environment in which a system or component is intended to be used*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.18

3.1985
operational ease GSC
1. one of the 14 general system characteristics describing the degree to which the application attends to operational aspects, such as, start-up, back-up, and recovery processes*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.1986
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)

3.1987
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

NOTE Operational scenarios are used to evaluate the requirements and design of the system and to verify and validate the system.

3.1988
operational testing
1. testing conducted to evaluate a system or component in its operational environment*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.21

cf. development testing, acceptance testing, qualification testing

3.1989
operations
1. ongoing execution of activities that produce the same product or provide a repetitive service

EXAMPLE production, manufacturing, accounting

3.1990
operator
1. a mathematical or logical symbol that represents an action to be performed in an operation. ***2.*** individual or organization that operates the system. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.40. ***3.*** entity that performs the operation of a system. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.22; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.13; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.30. ***4.*** a 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.*** an individual or an organization that contributes to the functionality of a system and draws on knowledge, skills, and procedures to contribute the function. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.25

NOTE The role of operator and the role of user may be vested, simultaneously or sequentially, in the same individual or organization. An individual operator combined with knowledge, skills and procedures may be considered as an element of the system. In the context of this specific definition, the term entity means an individual or an organization.

3.1991
operator manual
1. a document that provides the information necessary to initiate and operate a system or component. *Syn:* operator's manual, operations manual

cf. diagnostic manual, installation manual, programmer manual, support manual, user manual

NOTE Typically described are procedures for preparation, operation, monitoring, and recovery. An operator manual is distinguished from a user manual when a distinction is made between those who operate a computer system (mounting tapes, etc) and those who use the system for its intended purpose.

3.1992
opportunity
1. a condition or situation favorable to the project, a positive set of circumstances, a positive set of events, a risk that will have a positive impact on project objectives, or a possibility for positive changes. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. threat

3.1993
opportunity study
1. a study to examine a problem and determine whether or not it requires being solved during the time period under consideration. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.01

3.1994
optimizing process
1. quantitatively managed process that is improved based on an understanding of the common causes of variation inherent in the process

NOTE The focus of an optimizing process is on continually improving the range of process performance through both incremental and innovative improvements.

3.1995
optional
1. a syntax keyword used to specify a partial mapping. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.133

cf. mandatory, partial

3.1996
optional attribute
1. an attribute that may have no value for an instance. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.134

3.1997
optional category
1. category that provides additional details that are not essential but may be useful in particular situations. *IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies*.3.6

3.1998
optional nonidentifying relationship
1. a nonidentifying relationship in which an instance of the child entity can exist without being related to an instance of the parent entity. *IEEE Std 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]

3.1999
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 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998*.3.6

NOTE An optional requirement may be either: a) one of two or more alternative requirements, or b) an additional requirement that must be fulfilled only if applicable and may otherwise be disregarded. [ISO/IEC Guide 2]

3.2000
optional subgroup
1. subgroups which the user has the option of using during an elementary process that adds or creates an instance or the data. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2001
optional task
1. task that may be added to the minimum testing tasks to address specific requirements. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.1.22

3.2002
ORB
1. Object Request Broker. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.3

3.2003
ORB core
1. the ORB component which moves a request from a client to the appropriate adapter for the target object. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.2.10

3.2004
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

3.2005
ordinal scale
1. scale in which the measurement values are rankings

NOTE The assignment of defects to a severity level is a ranking.

3.2006
organization
1. person or a group of people and facilities with an arrangement of responsibilities, authorities and relationships*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.14. ***2.*** company, corporation, government, not-for-profit or other legally constituted body, including associations, clubs, partnerships, government agencies, publicly listed companies, private companies and sole traders, that has its own function(s) and administration*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.10. ***3.*** people and processes assembled to produce a specific output (product or service)*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection--Characterization of Interconnections.*3.9

NOTE An identified part of an organization (even as small as a single individual) or an identified group of organizations can be regarded as an organization if it has responsibilities, authorities and relationships.

3.2007
organization chart
1. graphical depiction of hierarchies and interrelationships among persons working together

3.2008
organization level
1. the management level or levels responsible for managing one or more data processing or information systems organizations*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2009
organizational breakdown structure (OBS)
1. [Tool] a hierarchically organized depiction of the project organization arranged so as to relate the work packages to the performing organizational units. (Sometimes OBS is written as Organization Breakdown Structure with the same definition.)*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2010
organizational maturity
1. extent to which an organization has explicitly and consistently deployed processes that are documented, managed, measured, controlled, and continually improved

NOTE Organizational maturity may be measured via appraisals.

3.2011
organizational policy
1. guiding principle typically established by senior management that is adopted by an organization to influence and determine decisions

3.2012
organizational process assets
1. [Output/Input] any or all process related assets, from any or all of the organizations involved in the project that are or can be used to influence the project's success. These process assets include formal and informal plans, policies, procedures, and guidelines. The process assets also include the organizations' knowledge bases such as lessons learned and historical information*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth 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 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.

3.2013
organizational unit
1. the part of an organization that is the subject of measurement*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process*.3.31. ***2.*** that part of an organization that is assessed*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.25

NOTE An organizational unit deploys one or more processes that operate within a coherent set of business goals. An organizational unit is typically part of a larger organization, although in a small organization the organizational unit may be the whole organization.

3.2014
origin
1. the address of the initial storage location assigned to a computer program in main memory

cf. assembled origin, loaded origin, starting address

3.2015
origin attribute
1. the classification of software as either developed or nondeveloped

3.2016
original source statement
1. source statement that is obtained from an external product

3.2017
orphan
1. line of text on its own at the end of a page*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.33

3.2018
OSE
1. Open System Environment*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2019
OSF
1. Open Software Foundation*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2020
OSI
1. Open Systems Interconnection*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2021
outer cardinality
1. the 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

3.2022
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.*** the process by which an information processing system, or any of its parts, transfers data outside of that system or part. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.34. ***4.*** [Process Output] 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) — Fourth Edition.* ***5.*** in an IDEF0 model, that which is produced by a function. *IEEE Std 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

3.2023
output arc (of a transition)
1. an 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

3.2024
output argument
1. an argument that has not been specified as an input argument. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.136

cf. input argument

NOTE It is possible for an output argument to have no value at the time a request is made.

3.2025
output arrow
1. an arrow or arrow segment that expresses IDEF0 output. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.90

NOTE That is, an object type set whose instances are created by a function by transforming the function's input. The arrowtail of an output arrow is attached to the right side of a box.

3.2026
output assertion
1. a 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

3.2027
output place (of a transition)
1. a 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

3.2028
output primitive
1. primitive that includes source statements, function points, and documents

3.2029
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

3.2030
output sort
1. the 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. *Syn:* range sort

3.2031
overflow exception
1. an 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

3.2032
overhead time
1. the 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

3.2033
overlay
1. a 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.*** a 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

3.2034
overlay supervisor
1. a routine that controls the sequencing and positioning of overlays

3.2035
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

3.2036
override
1. the 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.137

3.2037
overriding property
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.138

3.2038
owned attribute
1. an attribute of an entity that has not migrated into the entity*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.139

NOTE [key style]

3.2039
owner
1. person or organization that owns the copyright for the Candidate FSM method*. ISO/IEC 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.7

3.2040
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

3.2041
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

3.2042
package
1. a separately compilable software component consisting of related data types, data objects, and subprograms

cf. data abstraction, encapsulation, information hiding

3.2043
packaging
1. in software development, the assignment of modules to segments to be handled as distinct physical units for execution by a computer

3.2044
padding
1. the 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

3.2045
page
1. a 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.*** a screenful of information on a video display terminal

cf. paging

3.2046
page breakage
1. a 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

3.2047
page frame
1. a block of main storage having the size of, and used to hold, a page

cf. paging

3.2048
page reference
1. an expression that unambiguously identifies a model page*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.91

NOTE The page reference incorporates a diagram reference to the associated diagram, the type of page, and any sequencing data needed to distinguish different pages of the same type that are associated with the same diagram.

3.2049
page swapping
1. the exchange of pages between main storage and auxiliary storage

cf. paging

3.2050
page table
1. a table that identifies the location of pages in storage and gives significant attributes of those pages

cf. paging

3.2051
page type letter
1. the uppercase letter in a page reference that denotes a specific type of model page*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.92

3.2052
page zero
1. in the paging method of storage allocation, the first page in a series of pages

3.2053
pager
1. a routine that initiates and controls the transfer of pages between main and auxiliary storage

cf. paging

3.2054
paging
1. a 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.*** a 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.*** the transfer of pages as in (2). *Syn:* block allocation

cf. contiguous allocation, page, page breakage, page frame, page swapping, page table, page zero, pager, working set

3.2055
paper documentation
1. that part of the documentation which is in printed form*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.34

3.2056
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

3.2057
parallel classes
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.140

3.2058
parallel construct
1. a program construct consisting of two or more procedures that can occur simultaneously

3.2059
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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.04.05

3.2060
parameter
1. a variable that is given a constant value for a specified application. ***2.*** a constant, variable, or expression that is used to pass values between software modules. ***3.*** a 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

3.2061
parameterized collection class
1. a collection class restricted to hold only instances of a specified type (class)*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.141

3.2062
parameterized high-level net graph
1. a 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

3.2063
parametric estimating
1. [Technique] an estimating technique that uses a statistical relationship between historical data and other variables (e.g., square footage in construction, lines of code in software development) to calculate an estimate for activity parameters, such as scope, cost, budget, and duration. An example for the cost parameter is multiplying the planned quantity of work to be performed by the historical cost per unit to obtain the estimated cost*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2064
parent box
1. an ancestral box related to its child diagram by exactly one parent/child relationship*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.93

NOTE That is, a box detailed by a child diagram. The existence of this child diagram is indicated by a box detail reference.

3.2065
parent diagram
1. a diagram that contains a parent box*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.94

3.2066
parent entity
1. an entity in a specific relationship whose instances can be related to a number of instances of another entity (child entity)*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.142

NOTE [key style]

3.2067
parent function
1. a function modeled by a parent box*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0.*2.1.95

3.2068
Pareto chart
1. [Tool] a 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) — Fourth Edition*

3.2069
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

3.2070
parser
1. a software tool that parses computer programs or other text, often as the first step of assembly, compilation, interpretation, or analysis

3.2071
partial
1. an incomplete mapping*. IEEE Std 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 That is, some instances map to no related instance. An attribute may 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 may have no meaning for some instances, i.e., it may not give an answer or produce a response.

3.2072
partial cluster
1. a subclass cluster in which an instance of the superclass may exist without also being an instance of any of the subclasses*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.144. *Syn:* incomplete cluster

cf. total cluster, superclass

3.2073
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

3.2074
participant property
1. a property of a state class that reflects that class' knowledge of a relationship in which instances of the class participate*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.145

NOTE When a relationship exists between two state classes, each class contains a participant property for that relationship. A participant property is a mapping from a state class to a related (not necessarily distinct) state class. The name of each participant property is the name of the role that the other class plays in the relationship, or it may simply be 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.

3.2075
partitioning
1. decomposition; the separation of the whole into its parts

3.2076
party
1. organization entering into an agreement*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.15. ***2.*** an 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 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.5.1

EXAMPLE enterprise objects representing natural persons, legal entities, governments and their parts, and other associations or groups of natural persons Parties are responsible for their actions and the actions of their agents

3.2077
pass
1. a 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

3.2078
pass/fail criteria
1. decision rules used to determine whether a software item or a software feature passes or fails a test. *Syn:* pass-fail criteria

3.2079
passive I/O device
1. a device that does not generate an interrupt on completion of an input or output operation

NOTE The input from a passive input device must be read either on a polled basis or on demand.

3.2080
passive I/O device interface task
1. a task that interfaces to a passive I/O device and either reads from it or writes to it on demand

3.2081
passive interconnection
1. an interoperability agreement describing a common interpretation of one or more phenomena shared between two interacting things*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.1

3.2082
passive object
1. an object with no thread of control

NOTE an object with operations that concurrent objects (that is, tasks) invoke directly or indirectly

3.2083
passive white space
1. top, bottom, left and right margins which surround text. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.55

3.2084
patch
1. a modification made directly to an object program without reassembling or recompiling from the source program. ***2.*** a modification made to a source program as a last-minute fix or afterthought. ***3.*** a modification to a source or object program. ***4.*** to perform a modification as in (1), (2), or (3)

3.2085
path
1. in software engineering, a sequence of instructions that may be 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

3.2086
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

3.2087
path condition
1. a set of conditions that must be met in order for a particular program path to be executed

3.2088
path convergence
1. the merging or joining of parallel schedule network paths into the same node in a project schedule network diagram. Path convergence is characterized by a schedule activity with more than one predecessor activity. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2089
path divergence
1. extending or generating parallel schedule network paths from the same node in a project schedule network diagram. Path divergence is characterized by a schedule activity with more than one successor activity. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2090
path expression
1. a logical expression indicating the input conditions that must be met in order for a particular program path to be executed

3.2091
path testing
1. testing designed to execute all or selected paths through a computer program

cf. branch testing, statement testing

3.2092
pathological coupling
1. a 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

3.2093
pause
1. to suspend the execution of a computer program. *Syn:* halt (2)

cf. stop

3.2094
PCA
1. physical configuration audit

3.2095
PCO
1. Point of Control and Observation. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2096
PDL
1. program design language

3.2097
PDM
1. precedence diagramming method. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2098
PDR
1. preliminary design review

3.2099
peer review
1. review of work products performed by peers during development of the work products to identify defects for removal

cf. inspection, structured walkthrough, work product

3.2100
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) — Fourth Edition*

3.2101
perfective maintenance
1. modification of a software product after delivery to detect and correct latent faults in the software product before they are manifested as failures. *ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance*.3.7. ***2.*** software maintenance performed to improve the performance, maintainability, or other attributes of a computer program. ***3.*** improvements in software's performance or functionality, for example, in response to user suggestions and requests

NOTE Perfective maintenance provides enhancements for users, improvement of program documentation, and recoding to improve software performance, maintainability, or other software attributes.

3.2102
perform integrated change control
1. [Process] the process of reviewing all change requests, approving changes, and managing changes to the deliverables, organizational process assets, project documents, and project management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2103
perform qualitative analysis
1. [Process] 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) — Fourth Edition*

3.2104
perform quality assurance
1. [Process] the process of auditing the quality requirements and the results from quality control measurements to ensure appropriate quality standards and operational definitions are used. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2105
perform quality control
1. [Process] 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) — Fourth Edition*

3.2106
perform quantitative analysis
1. [Process] 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) — Fourth Edition*

3.2107
performance
1. the degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage

3.2108
performance analysis
1. a quantitative analysis of a real-time system (or software design) executing on a given hardware configuration with a given external workload applied to it

3.2109
performance GSC
1. one of the 14 general system characteristics describing the degree to which response time and throughput performance considerations influenced the application development. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2110
performance indicator
1. an assessment indicator that supports the judgment of the process performance of a specific process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.26

3.2111
performance measurement baseline
1. an approved integrated scope-schedule-cost plan for the project work against which project execution is compared to measure and manage performance. Technical and quality parameters may also be included. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2112
performance reports
1. [Output/Input] documents and presentations that provide organized and summarized work performance information, earned value management parameters and calculations, and analyses of project work progress and status. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2113
performance requirement
1. the measurable criterion that identifies a quality attribute of a function or how well a functional requirement must be accomplished. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.26. ***2.*** a system or software requirement specifying a performance characteristic that a system/software system or system/software component must possess ***3.*** a requirement that imposes conditions on a functional requirement

3.2114
performance specification
1. a document that specifies the performance characteristics that a system or component must possess

NOTE often part of a requirements specification. These characteristics typically include speed, accuracy, and memory usage.

3.2115
performance testing
1. testing conducted to evaluate the compliance of a system or component with specified performance requirements

cf. functional testing

3.2116
performed process
1. process that accomplishes the needed work to produce work products

NOTE satisfies the specific goals of the process area

3.2117
performing organization
1. the enterprise whose personnel are most directly involved in doing the work of the project. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2118
periodic I/O device interface task
1. a task that interfaces to a passive I/O device and polls it regularly

3.2119
periodic task
1. a task that a timer event activates at regular intervals

3.2120
peripheral equipment
1. a device that is controlled by and can communicate with a particular computer. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.07

EXAMPLE external storage

3.2121
permanence
1. the degree to which failures can affect object state changes due to completed transactions. *ISO/IEC 10746-3:1996 Information technology — Open Distributed Processing — Reference Model: Architecture.*13.7.1.5

3.2122
persistence schema
1. a specification of constraints on the use of specific processing, storage and communication functions*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*16.5.1.1

3.2123
persistence transparency
1. a distribution transparency which masks, from an object, the deactivation and reactivation of other objects (or itself)*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.7

NOTE Deactivation and reactivation are often used to maintain the persistence of an object when a system is unable to provide it with processing, storage and communication functions continuously.

3.2124
persistent storage
1. storage which enables a functional process to store data beyond the life of the functional process and/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*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.19

3.2125
persistent URI
1. 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 may have changed locations*. IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle*.3.1.9

3.2126
personal computer (PC)
1. a microcomputer primarily intended for stand-alone use by an individual*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.20

3.2127
personnel
1. an individual expected to perform duties on behalf of the organization, including officers, employees and contractors*. ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 8

3.2128
PERT
1. program evaluation and review technique

3.2129
Petri net
1. an 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.*** an abstract, formal model of information flow, showing static and dynamic properties of a system

NOTE A Petri net is usually represented as a graph having two types of nodes (called places and transitions) connected by arcs, and markings (called tokens) indicating dynamic properties.

3.2130
physical configuration audit (PCA)
1. an audit conducted to verify that a configuration item, as built, conforms to the technical documentation that defines it

cf. functional configuration audit

3.2131
physical requirement
1. a 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

3.2132
physical source statement (PSS)
1. source statement considered as a line of code

cf. logical source statement

3.2133
picture
1. illustration that shows the actual appearance of physical objects. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.32

EXAMPLE photograph, drawing

3.2134
PIGS
1. Protocol Implementation Conformance Statement. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*4

3.2135
pilot project
1. a 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 2382-20:1990 Information technology — Vocabulary — Part 20: System development.*20.01.07

3.2136
pipeline
1. a 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

3.2137
pixel
1. smallest element of a screen display; short for 'picture element'. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.35

3.2138
PIXIT
1. Protocol Implementation Extra Information for Testing. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*4

3.2139
place
1. a 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 A place is typed.

3.2140
place type
1. a 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 This set can describe an arbitrarily complex data structure.

3.2141
place/transition net
1. a 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

3.2142
plan communications
1. [Process] the process of determining project stakeholder information needs and defining a communication approach*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2143
plan procurement
1. [Process] the process of documenting project purchasing decisions, specifying the approach, and identifying potential sellers*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2144
plan quality
1. [Process] the process of identifying quality requirements and/or standards for the project and product, and documenting how the project will demonstrate compliance*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2145
plan risk management
1. [Process] the process of defining how to conduct risk management activities for a project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2146
plan risk responses
1. [Process] 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) — Fourth Edition*

3.2147
plan standard
1. a standard that describes the characteristics of a scheme for accomplishing defined objectives or work within specified resources

3.2148
planned process
1. process that is documented by both a description and a plan

NOTE The related process description and plan should be coordinated, and the plan should include standards, requirements, objectives, resources, and assignments.

3.2149
planned value (PV)
1. the authorized budget assigned to the scheduled work to be accomplished for a schedule activity or work breakdown structure component*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* budgeted cost of work scheduled (BCWS)

3.2150
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) — Fourth Edition*

cf. control account

3.2151
planning processes
1. [Process Group] those processes performed to establish the total scope of the effort, define and refine the project objectives, and develop the course of action required to attain those objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2152
platform
1. a collection of hardware and software components that are needed for a CASE tool to operate*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.11

3.2153
PM
1. Project Manager*. ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5. ***2.*** program manager

3.2154
PMBOK®
1. Project Management Body of Knowledge*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2155
PMIS
1. project management information system*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2156
PMO
1. project management office. ***2.*** program management office

3.2157
PMP®
1. Project Management Professional*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2158
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

3.2159
point
1. measure of vertical distance; there are approximately 28 points to the millimeter (72 points to the inch)*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.36

3.2160
point design
1. the selection of one design that satisfies the requirements without examining other, potentially more effective, designs

3.2161
pointer
1. a 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

3.2162
policy
1. a set of rules related to a particular purpose*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.4.1. ***2.*** clear and measurable statements of preferred direction and behavior to condition the decisions made within an organization*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.11

NOTE A rule can be expressed as an obligation, an authorization, a permission, or a prohibition. Not every policy is a constraint. Some policies represent an empowerment.

3.2163
portability
1. the ease with which a system or component can be transferred from one hardware or software environment to another. ***2.*** the 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.06. *Syn:* transportability

cf. machine-independent

3.2164
portable computer
1. a microcomputer that can be hand-carried for use in more than one location*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.21

3.2165
portfolio
1. a collection of projects or programs and other work that are grouped together to facilitate effective management of that work to meet strategic business objectives. The projects or programs of the portfolio may not necessarily be interdependent or directly related*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2166
portfolio management
1. [Technique] the centralized management of one or more portfolios, which includes identifying, prioritizing, authorizing, managing, and controlling projects, programs, and other related work, to achieve specific strategic business objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2167
port-to-port time
1. the 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

3.2168
postcondition
1. a condition that is guaranteed to be true after a successful property request*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.147. ***2.*** a constraint that must be true when a use case has ended. *Syn:* post-condition

3.2169
postmortem dump
1. a dump that is produced upon abnormal termination of a computer program. *Syn:* post-mortem dump

cf. change dump, dynamic dump, memory dump, selective dump, snapshot dump, static dump

3.2170
postprocessor
1. a computer program or routine that carries out some final processing step after the completion of the primary process. *Syn:* post-processor

cf. preprocessor

EXAMPLE a routine that reformats data for output

3.2171
powertype
1. a type the instances of which are subtypes of the partitioned type. *ISO/IEC 24744:2007, 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

3.2172
PR&RPI
1. Problem Reporting and Resolution Planned Information. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2173
practice
1. an activity that contributes to the purpose or outcomes of a process or enhances the capability of a process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.27. ***2.*** requirements employed to prescribe a disciplined uniform approach to the software development process. ***3.*** a specific type of professional or management activity that contributes to the execution of a process and that may employ one or more techniques and tools. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. conventions, standards

3.2174
precedence diagramming method (PDM)
1. [Technique] a schedule network diagramming technique in which schedule activities are represented by boxes (or nodes). Schedule activities 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) — Fourth Edition. Syn:* activity-on-node (AON)

3.2175
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) — Fourth Edition*

cf. logical relationship

3.2176
precision
1. the degree of exactness or discrimination with which a quantity is stated

cf. accuracy

EXAMPLE a precision of 2 decimal places versus a precision of 5 decimal places

3.2177
precompiler
1. a computer program or routine that processes source code and generates equivalent code that is acceptable to a compiler

EXAMPLE a routine that converts structured FORTRAN to ANSI-standard FORTRAN

3.2178
precondition
1. a condition that is required to be true before making a property request. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.148.
2. a constraint that must be true when a use case is invoked. *Syn:* pre-condition

3.2179
predecessor activity
1. the schedule activity that determines when the logical successor activity can begin or end. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2180
predictive metric
1. a metric applied during development and used to predict the values of a software quality factor. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.13

3.2181
predictive metric value
1. a numerical target related to a quality factor to be met during system development. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.14

NOTE This is an intermediate requirement that is an early indicator of final system performance. For example, design or code errors may be early predictors of final system reliability.

3.2182
preliminary design
1. the process of analyzing design alternatives and defining the architecture, components, interfaces, and timing and sizing estimates for a system or component ***2.*** the result of the process in (1)

cf. detailed design

3.2183
preliminary design review (PDR)
1. a 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.*** a review as in (1) of any hardware or software component

cf. critical design review, system design review

3.2184
preparation time
1. the 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 The event of starting the preparation time depends on the definition of the task mode of the following task.

3.2185
preprocessor
1. a 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

3.2186
prescription
1. an action that establishes a rule*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.5.6

3.2187
presentable
1. can be retrieved and viewed*. IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*H.3 g)

3.2188
prestore
1. to store data that are required by a computer program or routine before the program or routine is entered

3.2189
prettyprinting
1. the use of indentation, blank lines, and other visual cues to show the logical structure of a program

3.2190
preventive action
1. a documented direction to perform an activity that can reduce the probability of negative consequences associated with project risks*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2191
preventive maintenance
1. the modification of a software product after delivery to detect and correct latent faults in the software product before they become operational faults*. ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.8. ***2.*** maintenance performed for the purpose of preventing problems before they occur. ***3.*** changes to hardware or software performed to prevent future defects or failures*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***4.*** continuously upgrading a system to enable it to cope with current and future changes. ***5.*** designing a software system that is easy to maintain

3.2192
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 Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.2

3.2193
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

3.2194
primary key
1. the candidate key selected as the unique identifier of an entity. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.149

NOTE [key style]

3.2195
primitive
1. the 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 Primitives are directly measurable or countable, or may be given a constant value or condition for a specific measure.

3.2196
principal
1. a party that has delegated (authority, a function, etc) to another. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.5.8

3.2197
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.33

cf. embedded documentation

3.2198
priority
1. the level of importance assigned to an item. ***2.*** a rank order of status, activities, or tasks. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document*.3.1

3.2199
priority ceiling protocol
1. an algorithm that provides bounded priority inversion

NOTE that is, at most one lower-priority task can block a higher-priority task

3.2200
priority interrupt
1. an interrupt performed to permit execution of a process that has a higher priority than the process currently executing

3.2201
priority inversion
1. a case where a task's execution is delayed because a lower priority task is blocking it

3.2202
private
1. a responsibility that is visible only to the class or the receiving instance of the class (available only within methods of the class). *IEEE Std 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

3.2203
private type
1. a data type whose structure and possible values are defined but are not revealed to the user of the type

cf. information hiding

3.2204
privileged instruction
1. a computer instruction that can be executed only by a supervisory program

3.2205
probability
1. the extent to which an event is likely to occur*. ISO/IEC 16085:2006 (IEEE Std 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 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.3

NOTE For a high degree of belief, the probability is near 1. Frequency rather than probability may 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. [ISO Guide 73:2002, definition 3.1.3]

3.2206
probability and impact matrix
1. [Tool] a common way to determine whether a risk is considered low, moderate, or high by combining the two dimensions of a risk: its probability of occurrence, and its impact on objectives if it occurs*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2207
problem
1. unknown underlying cause of one or more incidents*. ISO/IEC 20000-1:2005, nformation technology — Service management — Part 1: Specification.*2.8***2.*** a negative situation to overcome

NOTE risk factor becomes a problem when a risk metric (an objective measure) crosses a predetermined threshold (the problem trigger).

3.2208
problem definition
1. a statement of a problem, which may include a description of the data, the method, the procedures, and algorithms used to solve it*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.03. *Syn:* problem description

3.2209
problem domain
1. a set of similar problems that occur in an environment and lend themselves to common solutions*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2

3.2210
problem report (PR)
1. a document used to identify and describe problems detected in a product*. ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.9

NOTE PRs are either submitted directly to denote faults or established after impact analysis is performed on Modification Requests and faults are found.

3.2211
problem state
1. in the operation of a computer system, a state in which programs other than the supervisory program can execute. *Syn:* slave state, user state

cf. supervisor state

3.2212
problem-oriented language
1. a programming language designed for the solution of a given class of problems

EXAMPLE list processing languages, information retrieval languages, simulation languages

3.2213
procedural cohesion
1. a 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

3.2214
procedural language
1. a programming language in which the user states a specific set of instructions that the computer must perform in a given sequence. *Syn:* procedure-oriented language

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

NOTE All widely-used programming languages are of this type.

3.2215
procedural programming language
1. a computer programming language used to express the sequence of operations to be performed by a computer. *IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

cf. nonprocedural programming language

EXAMPLE COBOL

3.2216
procedure
1. ordered series of steps that specify how to perform a task. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.34. ***2.*** a written description of a course of action as in (1). ***3.*** a portion of a computer program that is named and that performs a specific action. ***4.*** specified way to carry out an activity or process. *ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 9. ***5.*** a routine that does not return a value

3.2217
process
1. set of interrelated or interacting activities which transforms inputs into outputs. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.16; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.32. ***2.*** a predetermined course of events defined by its purpose or by its effect, achieved under given conditions. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.24. ***3.*** to perform operations on data. ***4.*** a collection of steps taking place in a prescribed manner and leading to an objective. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.3.5. ***5.*** in data processing, the predetermined course of events that occur during the execution of

all or part of a program. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.25. ***6.*** an executable unit managed by an operating system scheduler. ***7.*** system of activities, which use resources to transform inputs into outputs. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.41

NOTE [ISO 9000:2005] The term ""activities"" covers use of resources. A process may have multiple starting points and multiple end points. The prescribed manner may be a partially ordered sequence. A process specification can be a workflow specification. An enterprise specification may define types of processes and may define process templates.

3.2218
process action plan
1. plan, usually resulting from appraisals, that documents how specific improvements targeting the weaknesses uncovered by an appraisal will be implemented

3.2219
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

3.2220
process architect
1. the person or group that has primary responsibility for creating and maintaining the software life cycle process (SLCP). *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*Annex E

cf. software life cycle process (SLCP)

3.2221
process architecture
1. ordering, interfaces, interdependencies, and other relationships among the process elements in a standard process

NOTE Process architecture also describes the interfaces, interdependencies, and other relationships between process elements and external processes, such as contract management.

3.2222
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

3.2223
process assessment
1. a disciplined evaluation of an organizational unit's processes against a Process Assessment Model. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.29

3.2224
process assessment model
1. a model suitable for the purpose of assessing process capability, based on one or more process reference models. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.30

3.2225
process asset library
1. collection of information that can be useful to those who are defining, implementing, and managing processes in the organization

NOTE may include process-related documentation such as policies, defined processes, checklists, lessons-learned documents, templates, standards, procedures, plans, and training materials

3.2226
process attribute
1. a measurable characteristic of process capability applicable to any process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.31

3.2227
process attribute rating
1. a judgment of the degree of achievement of the process attribute for the assessed process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.32

3.2228
process capability
1. a characterization of the ability of a process to meet current or projected business goals. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.33.
2. range of expected results that can be achieved by following a process

3.2229
process capability determination
1. a systematic assessment and analysis of selected processes within an organization against a target capability, carried out with the aim of identifying the strengths, weaknesses and risks associated with deploying the processes to meet a particular specified requirement. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.34

3.2230
process capability determination sponsor
1. the individual or entity, internal or external to the organizational unit being assessed, who requires the process capability determination to be performed, and provides financial or other resources to carry it out. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.35

3.2231
process capability level
1. a point on the six-point ordinal scale (of process capability) that represents the capability of the process; each level builds on the capability of the level below. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.36

3.2232
process capability level rating
1. a representation of the achieved process capability level derived from the process attribute ratings for an assessed process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.37

3.2233
process context
1. the set of factors, documented in the assessment input, that influence the judgment, comprehension and comparability of process attribute ratings. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.38

3.2234
process description
1. documented expression of a set of activities performed to achieve a given purpose

NOTE A process description provides an operational definition of the major components of a process. The description specifies, in a complete, precise, and verifiable manner, the requirements, design, behavior, or other characteristics of a process. It also may include procedures for determining whether these provisions have been satisfied. Process descriptions can be found at the activity, project, or organizational level.

3.2235
process dimension
1. the set of elements in a process assessment model explicitly related to the processes defined in the relevant process reference model(s). *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.39

NOTE The processes may be grouped based on different criteria.

3.2236
process group
1. collection of related processes. ***2.*** team of specialists who facilitate the definition, maintenance, and improvement of processes used by the organization

3.2237
process improvement
1. actions taken to change an organization's processes so that they more effectively and/or efficiently meet the organization's business goals. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.40. ***2.*** the result of activities that better the performance and maturity of the organization's processes

3.2238
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

3.2239
process improvement program
1. the strategies, policies, goals, responsibilities and activities concerned with the achievement of specified improvement goals. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.41

NOTE A process improvement program can span more than one complete cycle of process improvement.

3.2240
process improvement project
1. a subset of the process improvement program that forms a coherent set of actions to achieve a specific improvement. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.42

3.2241
process improvement sponsor
1. the individual or entity, internal or external to the organizational unit being assessed, who requires the process improvement to be performed, and provides financial or other resources to carry it out. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.43

3.2242
process management
1. the direction, control, and coordination of work performed to develop a product or perform a service

EXAMPLE quality assurance

3.2243
process measures
1. information captured about the development process. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2244
process metric
1. a metric used to measure characteristics of the methods, techniques, and tools employed in developing, implementing, and maintaining the software system. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.15

3.2245
process outcome
1. an observable result of a process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.44. ***2.*** observable result of the successful achievement of the process purpose. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.27; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.18

NOTE 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.

3.2246
process owner
1. person (or team) responsible for defining and maintaining a process

NOTE At the organizational level, the process owner is the person (or team) responsible for the description of a standard process; at the project level, the process owner is the person (or team) responsible for the description of the defined process. A process may therefore have multiple owners at different levels of responsibility.

3.2247
process performance
1. the extent to which the execution of a process achieves its purpose. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.45

3.2248
process profile
1. the set of process attribute ratings for an assessed process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.46

3.2249
process purpose
1. high-level objective of performing the process and the likely outcomes of effective implementation of the process. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.17. ***2.*** the high-level measurable objectives of performing the process and the likely outcomes of effective implementation of the process. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.47

NOTE The implementation of the process should provide tangible benefits to the stakeholders.

3.2250
process reference model
1. a model comprising definitions of processes in a life cycle described in terms of process purpose and outcomes, together with an architecture describing the relationships between the processes. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.48

3.2251
process standard
1. a standard that deals with the series of actions or operations used in making or achieving a product

3.2252
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

3.2253
processing logic
1. requirements specifically requested by the user to complete an elementary process. *ISO/IEC 20926:2003; Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*7

NOTE such as validations, algorithms, or calculations, and reading or maintaining a file

3.2254
processor
1. in a computer, a functional unit that interprets and executes instructions. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.08

NOTE A processor consists of at least an instruction control unit and an arithmetic and logic unit.

3.2255
procurement documents
1. [Output/Input] those 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) — Fourth Edition*

3.2256
procurement management plan
1. [Output/Input] the document that describes how procurement processes from developing procurement documentation through contract closure will be managed. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2257
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) — Fourth Edition.* ***2.*** complete set of software and documentation. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.36. ***3.*** output of the software development activities (e.g., document, code, or model). *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*Annex E. ***4.*** result of a process. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.33. *Syn:* material, goods

cf. activity, deliverable, result

NOTE [ISO 9000:2005] There are four agreed generic product categories: 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.

3.2258
product analysis
1. the process of evaluating a product by manual or automated means to determine if the product has certain characteristics

3.2259
product authority
1. person or persons with overall responsibility for the capabilities and quality of a product. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.37

3.2260
product baseline
1. in configuration management, the initial approved technical documentation (including, for software, the source code listing) defining a configuration item during the production, operation, maintenance, and logistic support of its life cycle ***2.*** the hardware/software configuration identification established at the end of a development phase

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

3.2261
product configuration identification
1. the current approved or conditionally approved technical documentation defining a configuration item during the production, operation, maintenance, and logistic support phases of its life cycle

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

NOTE It prescribes all necessary physical or form, fit, and function characteristics of a configuration item, the selected functional characteristics designated for production acceptance testing, and the production

3.2262
product description
1. a description of the properties of a software package, with the main purpose of helping potential buyers to evaluate for themselves the suitability of the product before purchasing it

NOTE The product description is not a specification; it serves a different purpose.

3.2263
product engineering
1. the technical processes to define, design, and construct or assemble a product

3.2264
product life cycle
1. a collection of generally sequential, non-overlapping product phases whose name and number are determined by the manufacturing and control needs of the organization. The last product life cycle phase for a product is generally the product's retirement. Generally, a project life cycle is contained within one or more product life cycles*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2265
product line
1. group of products or services sharing a common, managed set of features that satisfy specific needs of a selected market or mission ***2.*** a collection of systems that are potentially derivable from a single domain architecture*. IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.14

3.2266
product management
1. the definition, coordination, and control of the characteristics of a product during its development cycle

EXAMPLE configuration management

3.2267
product measures
1. information captured about the developed software application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2268
product metric
1. a metric used to measure the characteristics of any intermediate or final product of the software development process*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.16

3.2269
product requirement
1. refinement of customer requirements into the developers' language, making implicit requirements into explicit derived requirements

3.2270
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) — Fourth Edition*

3.2271
product scope description
1. the documented narrative description of the product scope*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2272
product specification
1. a document that specifies the design that production copies of a system or component must implement.
2. a document that describes the characteristics of a planned or existing product for consideration by potential customers or users

cf. design description

NOTE For software, this document describes the as-built version of the software.

3.2273
product standard
1. a standard that defines what constitutes completeness and acceptability of items that are used or produced, formally or informally, during the software engineering process

3.2274
product support
1. the providing of information, assistance, and training to install and make software operational in its intended environment and to distribute improved capabilities to users

3.2275
production
1. steps involved in taking draft text and turning it into camera-ready originals, completed help text or online documentation*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.39

3.2276
production library
1. a software library containing software approved for current operational use

cf. master library, software development library, software repository, system library

3.2277
productivity
1. the ratio of work product to work effort*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. delivery rate

3.2278
professional standard
1. a standard that identifies a profession as a discipline and distinguishes it from other professions

3.2279
program
1. to write a computer program. ***2.*** to design, write, modify, and test programs*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.02. ***3.*** a group of related projects managed in a coordinated way to obtain benefits and control not available from managing them individually. Programs may include elements of related work outside of the scope of the discrete projects in the program*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. computer program

3.2280
program construct
1. a set of one or more procedure parts and a control part which may be implicit*. ISO/IEC 8631:1989, Information technology — Program constructs and conventions for their representation.*2

NOTE Each procedure part consists of one or more operations to be performed or may be null.

3.2281
program design language (PDL)
1. a specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document a program design

cf. hardware design language, pseudo code

3.2282
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) — Fourth Edition*

3.2283
program instruction
1. a computer instruction in a source program

NOTE A program instruction is distinguished from a computer instruction that results from assembly, compilation, or other interpretation process.

3.2284
program librarian
1. the person responsible for establishing, controlling, and maintaining a software development library

3.2285
program listing
1. a printout or other human readable display of the source and, sometimes, object statements that make up a computer program

3.2286
program maintenance manual
1. a document that provides the information necessary to maintain a program*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.08

3.2287
program management
1. the centralized coordinated management of a program to achieve the program's strategic objectives and benefits*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2288
program mutation
1. a computer program that has been purposely altered from the intended version to evaluate the ability of test cases to detect the alteration ***2.*** the process of creating an altered program as in (1)

cf. mutation testing

3.2289
program network chart
1. a diagram that shows the relationship between two or more computer programs

3.2290
program specification
1. a document that describes the structure and functions of a program in sufficient detail to permit programming and to facilitate maintenance. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.06

3.2291
program status word (PSW)
1. a computer word that contains information specifying the current status of a computer program. ***2.*** a special-purpose register that contains a program status word as in (1)

NOTE The information may include error indicators, the address of the next instruction to be executed, currently enabled interrupts, and so on.

3.2292
program synthesis
1. the use of software tools to aid in the transformation of a program specification into a program that realizes that specification

3.2293
programmable breakpoint
1. a breakpoint that automatically invokes a previously specified debugging process when initiated

cf. code breakpoint, data breakpoint, dynamic breakpoint, epilog breakpoint, prolog breakpoint, static breakpoint

3.2294
programmable terminal
1. a user terminal that has built-in data processing capability. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.14. *Syn:* intelligent terminal

3.2295
programmer manual
1. a 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 Typically described are the equipment configuration, operational characteristics, programming features, input/output features, and compilation or assembly features of the computer system.

3.2296
programming
1. the general activity of software development. ***2.*** the designing, writing, modifying, and testing of programs. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.03

cf. construction

NOTE especially construction activities

3.2297
programming language
1. a language used to express computer programs. ***2.*** an artificial language for expressing programs. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.10

3.2298
programming support environment
1. an integrated collection of software tools accessed via a single command language to provide programming support capabilities throughout the software life cycle

cf. scaffolding

NOTE sometimes called integrated programming support environment. The environment typically includes tools for specifying, designing, editing, compiling, loading, testing, configuration management, and project management.

3.2299
programming system
1. a set of programming languages and the support software (editors, compilers, linkers, etc.) necessary for using these languages with a given computer system

3.2300
program-sensitive fault
1. a fault that causes a failure when some particular sequence of program steps is executed

cf. data-sensitive fault

3.2301
progressive elaboration
1. [Technique] continuously improving and detailing a plan as more detailed and specific information and more accurate estimates become available as the project progresses, and thereby producing more accurate and complete plans that result from the successive iterations of the planning process. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2302
project
1. endeavor with defined start and finish dates undertaken to create a product or service in accordance with specified resources and requirements. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.20; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.34. ***2.*** an undertaking with pre-specified objectives, magnitude and duration. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.01. ***3.*** a temporary endeavor undertaken to create a unique product, service, or result. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***4.*** a collection of work tasks with a time frame and a work product to be delivered. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***5.*** set of activities for developing a new product or enhancing an existing product. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.38

NOTE A project may be viewed as a unique process comprising coordinated and controlled activities and may be composed of activities from the Project Processes and Technical Processes.

3.2303
project agreement
1. a document or set of documents baselined by the acquirer and the supplier that specifies the conditions under which the project will be conducted. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.4

NOTE A project agreement may include items such as the scope, objectives, assumptions, management interfaces, risks, staffing plan, resource requirements, price, schedule, resource and budget allocations, project deliverables, and acceptance criteria for the project deliverables. Documents in a project agreement may include some or all of the

following: a contract, a statement of work, user requirements, system engineering specifications, software requirements specifications, a software project management plan, supporting process plans, a business plan, a project charter, or a memo of understanding.

3.2304
project calendar
1. a calendar of working days or shifts that establishes those dates on which schedule activities are worked and nonworking days that determine those dates on which schedule activities are idle. Typically defines holidays, weekends and shift hours*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. resource calendar

3.2305
project charter
1. [Output/Input] 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) — Fourth Edition*

3.2306
project communications management
1. [Knowledge Area] project communications management includes the processes required to ensure timely and appropriate generation, collection, distribution, storage, retrieval, and ultimate disposition of project information*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2307
project control
1. the activities concerned with monitoring the progress of a project, its direction, quality, and resource utilization, as compared with project plans*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.04

3.2308
project cost management
1. [Knowledge Area] project cost management includes the processes involved in estimating, budgeting, 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) — Fourth Edition*

3.2309
project deliverable
1. a work product to be delivered to the acquirer*. IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.5

NOTE Quantities, delivery dates, and delivery locations are specified in a project agreement. Project deliverables may include the following: operational requirements, functional specifications, design documentation, source code, object code, test results, installation instructions, training aids, user manuals, product development tools, and maintenance documentation. Project deliverables may be self-contained or may be part of a larger system's deliverables.

3.2310
project file
1. a central repository of material pertinent to a project. *Syn:* project notebook

NOTE Contents typically include memos, plans, technical reports, and related items.

3.2311
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 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.

3.2312
project human resource management
1. [Knowledge Area] project human resource management includes the processes that organize and manage the project team*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2313
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) — Fourth Edition*

3.2314
project integration management
1. [Knowledge Area] 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) — Fourth Edition*

3.2315
project leader
1. a person who manages or leads projects*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2316
project level
1. the management level responsible for managing individual new development or major enhancement projects*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2317
project life cycle
1. a collection of generally sequential project phases whose name and number are determined by the control needs of the organization or organizations involved in the project. A life cycle can be documented with a methodology*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2318
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) — Fourth Edition.*
2. the activities concerned with project planning and project control*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.02

3.2319
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) — Fourth Edition*

3.2320
project management information system (PMIS)
1. [Tool] 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) — Fourth Edition*

3.2321
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) — Fourth Edition*

3.2322
project management office (PMO)
1. an organizational body or entity assigned various responsibilities related to the centralized and coordinated management of those projects under its domain. The responsibilities of a PMO can range from providing project management support functions to actually being responsible for the direct management of a project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2323
project management plan
1. [Output/Input] a formal, approved document that defines how the project is executed, monitored and controlled. It may be a summary or detailed and may be composed of one or more subsidiary management plans and other planning documents*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2324
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) — Fourth Edition*

3.2325
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

3.2326
project management system
1. [Tool] 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) — Fourth Edition*

3.2327
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) — Fourth Edition*

3.2328
project manager (PM)
1. the person assigned by the performing organization to achieve the project objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** person who manages one or more projects or groups of projects*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***3.*** person with overall responsibility for the management and running of a project*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.39

3.2329
project organization chart
1. [Output/Input] 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) — Fourth Edition*

3.2330
project phase
1. a collection of logically related project activities, usually culminating in the completion of a major deliverable. Project phases are mainly completed sequentially, but can overlap in some project situations. A project phase is a component of a project life cycle. A project phase is not a project management process group*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2331
project plan
1. a document that describes the technical and management approach to be followed for a project

EXAMPLE a software development plan

NOTE 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.

3.2332
project planning
1. the activities concerned with the specification of the components, timing, resources, and procedures of a project*. ISO/IEC 2382-20:1990 Information technology--Vocabulary--Part 20: System development.*20.07.03

3.2333
project portfolio
1. collection of projects that addresses the strategic objectives of the organization*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.30; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.21

3.2334
project procurement management
1. [Knowledge Area] project procurement management includes the processes to purchase or acquire the products, services, or results needed from outside the project team to perform the work*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2335
project quality management
1. [Knowledge Area] 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) — Fourth Edition*

3.2336
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

3.2337
project risk management
1. [Knowledge Area] project risk management includes the processes concerned with conducting risk management planning, identification, analysis, responses, and monitoring and control on a project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2338
project risk profile
1. a 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 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.4

cf. risk profile, risk state

NOTE 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.

3.2339
project schedule
1. [Output/Input] the planned dates for performing schedule activities and the planned dates for meeting schedule milestones*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2340
project schedule network diagram
1. [Output/Input] any schematic display of the logical relationships among the project schedule activities. Always drawn from left to right to reflect project work chronology*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* logic diagram

3.2341
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) — Fourth Edition*

3.2342
project scope management
1. [Knowledge Area] 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) — Fourth Edition*

3.2343
project scope statement
1. [Output/Input] the narrative description of the project scope, including major deliverables, project assumptions, project constraints, and a description of work, that provides a documented basis for making future project decisions and for confirming or developing a common understanding of project scope among the stakeholders*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2344
project specification
1. a specification of the objectives, requirements, and scope of a project and its relations to other projects*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.07.07

3.2345
project team
1. the 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

3.2346
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) — Fourth Edition*

3.2347
project time management
1. [Knowledge Area] 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) — Fourth Edition*

3.2348
project/application attribute
1. characteristics of a project or an application that may have a significant impact on productivity*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

EXAMPLE hardware platform, personnel experience, tools, and methodology

NOTE The project/application attribute is used to categorize project data during analysis.

3.2349
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) — Fourth Edition*

3.2350
prolog breakpoint
1. a breakpoint that is initiated upon entry into a program or routine. *Syn:* preamble breakpoint

cf. epilog breakpoint, code breakpoint, data breakpoint, dynamic breakpoint, programmable breakpoint, static breakpoint

3.2351
prompt
1. a 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.*** a visual or audible message sent by a program to request or guide the user's response*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.07

3.2352
proof
1. final copy of a paper document presented to the acquirer for review prior to publication*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.40

NOTE Unless alterations are requested, the finished document should be identical to the proof copy in all respects other than paper stock, binding and colors. Proofs are generally photocopies of the camera-ready originals.

3.2353
proof of correctness
1. a formal technique used to prove mathematically that a computer program satisfies its specified requirements. ***2.*** a proof that results from applying the technique in (1)

cf. assertion, formal specification, inductive assertion method, partial correctness, total correctness

3.2354
property
1. a responsibility that is an inherent or distinctive characteristic or trait that manifests some aspect of an object's knowledge or behavior*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.151

NOTE Three kinds of property are defined: attribute, participant properties due to relationships, and operations.

3.2355
proposal
1. compilation of benefits, costs, risks, opportunities, and other factors applicable to decisions to be made*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.12

NOTE includes business cases

3.2356
proposed change
1. a 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 The disposition may be to reject, to defer for further analysis, or to accept. Upon acceptance the proposed change becomes an approved modification. There may be a one-to-one, one-to-many, or many-to-many relationship between proposed changes and approved modifications.

3.2357
protected
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.152

cf. private, public, hidden

3.2358
protection exception
1. an 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

3.2359
protocol
1. a set of conventions that govern the interaction of processes, devices, and other components within a system

3.2360
protocol object
1. an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.12

3.2361
prototype
1. a 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.*** model or preliminary implementation of a piece of software suitable for the evaluation of system design, performance or production potential, or for the better understanding of the software requirements*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.41

NOTE A prototype is used to get feedback from users for improving and specifying a complex human interface, for feasibility studies, or for identifying requirements.

3.2362
prototyping
1. a 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

3.2363
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 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.8

NOTE These types of provision are distinguished [in English] by the form of wording they employ, e.g., instructions are expressed in the imperative mood, recommendations by the use of the auxiliary "should", and requirements by the use of the auxiliary "shall". [ISO/IEC Guide 2:2004]

3.2364
provisional assessor
1. a person who has the skills and competencies to carry out assessments under the guidance and supervision of a competent assessor. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.49

3.2365
PSCI
1. Protocol Support for Computational Interactions. *ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*4

3.2366
pseudo instruction
1. a source language instruction that provides information or direction to the assembler or compiler and is not translated into a target language instruction. *Syn:* pragma, pseudo-op, pseudo operation

EXAMPLE an instruction specifying the desired format of source code listings

3.2367
pseudocode
1. a combination of programming language constructs and natural language used to express a computer program design. ***2.*** a 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.

3.2368
PSW
1. program status word

3.2369
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

3.2370
public
1. a responsibility that is not hidden. *IEEE Std 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 that is, visible to any requester (available to all without restriction)

3.2371
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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.96

3.2372
PV
1. planned value. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2373
QA
1. quality assurance

3.2374
QC
1. quality control

3.2375
QME
1. quality measure element. *ISO/IEC 25021:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements.*5

3.2376
QOS
1. Quality of Service. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2377
QR
1. quality requirements. *ISO/IEC TR 14143-4:2002, Information technology — Software measurement — Functional size measurement — Part 4: Reference model.*4

3.2378
qualification
1. process of demonstrating whether an entity is capable of fulfilling specified requirements. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.31; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.22. ***2.*** the process of determining whether a system or component is suitable for operational use

3.2379
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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.32

3.2380
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 (IEEE Std 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. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.24

cf. acceptance testing, development testing, operational testing

3.2381
qualitative risk analysis
1. prioritizing risks for subsequent further analysis or action by assessing and combining their probability of occurrence and impact

3.2382
quality
1. the degree to which a system, component, or process meets specified requirements. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.25. ***2.*** ability of a product, service, system, component, or process to meet customer or user needs, expectations, or requirements. ***3.*** the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs. *ISO/IEC 9126-1:2001, Software engineering — Product quality — Part 1: Quality model.*B.21. ***4.*** conformity to user expectations, conformity to user requirements, customer satisfaction, reliability, and level of defects present. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***5.*** the degree to which a set of inherent characteristics fulfils requirements. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***6.*** the degree to which a system, component, or process meets customer or user needs or expectations. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.25

3.2383
quality assurance (QA)
1. a planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements ***2.*** a set of activities designed to evaluate the process by which products are developed or manufactured. ***3.*** 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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.34. ***4.*** part of quality management focused on providing confidence that quality requirements will be fulfilled. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.23

NOTE There are both internal and external purposes for quality assurance: within an organization, quality assurance provides confidence to management; in contractual situations, quality assurance provides confidence to the customer or others. Some quality control and quality assurance actions are interrelated. Unless requirements for quality fully reflect the needs of the user, quality assurance may not provide adequate confidence.

3.2384
quality attribute
1. characteristic of software, or a generic term applying to quality factors, quality subfactors, or metric values. *IEEE Std 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

3.2385
quality control (QC)
1. a set of activities designed to evaluate the quality of developed or manufactured products. ***2.*** the process of verifying one's own work or that of a co-worker.

cf. quality assurance

NOTE This term has no standardized meaning in software engineering at this time.

3.2386
quality evaluation
1. systematic examination of the extent to which an entity is capable of fulfilling specified requirements. *ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.22

NOTE The requirements may 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.

3.2387
quality factor
1. a management-oriented attribute of software that contributes to its quality. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.18. ***2.*** higher-level quality attribute

3.2388
quality factor sample
1. a set of quality factor values that is drawn from the metrics database and used in metrics validation. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.19

3.2389
quality factor value
1. a value of the direct metric that represents a quality factor. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.2

cf. metric value

3.2390
quality in use (measure)
1. the 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 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.42

cf. usability

NOTE This definition of quality in use is similar to the definition of usability in ISO 9241-11.

3.2391
quality management
1. coordinated activities to direct and control an organization with regard to quality. *ISO/IEC TR 19759:2005; Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*12

NOTE ISO 9000-2000.

3.2392
quality management plan
1. [Output/Input] the quality management plan describes how the project management team will implement the performing organization's quality policy. The quality management plan is a component or a subsidiary plan of the project management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2393
quality measure element (QME)
1. base measure or derived measure that is used for constructing software quality measures. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.43; *ISO/IEC 25021:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements.*4.14

NOTE The software quality characteristics or subcharacteristics of the entity are derived afterwards by calculating a software quality measure.

3.2394
quality metric
1. a quantitative measure of the degree to which an item possesses a given quality attribute. ***2.*** a 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

3.2395
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 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.44. ***2.*** the set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.24

3.2396
quality requirement
1. a requirement that a software attribute be present in software to satisfy a contract, standard, specification, or other formally imposed document*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.21

3.2397
quality subfactor
1. a decomposition of a quality factor or quality subfactor to its technical components*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.22

3.2398
quantitative risk analysis
1. numerical analysis of the effect on overall project objectives of identified risks

3.2399
quantity (base)
1. one of the quantities that, in a system of quantities, are conventionally accepted as functionally independent of one another*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.2

3.2400
query language
1. a language used to access information stored in a database

cf. programming language, specification language

3.2401
queue
1. a list in which items are appended to the last position of the list and retrieved from the first position of the list

3.2402
quiescing
1. the process of bringing a device or system to a halt by rejecting new requests for work

3.2403
RACI
1. responsible, accountable, consult, and inform*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2404
RAM
1. responsibility assignment matrix*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2405
random failure
1. a failure whose occurrence is unpredictable except in a probabilistic or statistical sense

cf. intermittent fault, transient error

3.2406
rapid prototyping
1. a 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

3.2407
rate-monotonic algorithm
1. a real-time scheduling algorithm that assigns higher priorities to tasks with shorter periods

3.2408
rating
1. action of mapping the measured value to the appropriate rating level*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.45

NOTE used to determine the rating level associated with the software for a specific quality characteristic. Rating and rating levels can be applied to characteristics other than quality characteristics.

3.2409
rating interval
1. the 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

3.2410
rating level
1. scale point on an ordinal scale, which is used to categorize a measurement scale*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.46

NOTE The rating level enables software product to be classified (rated) in accordance with the stated or implied needs. Appropriate rating levels may be associated with the different views of quality, i.e., Users', Managers,' or Developers.'

3.2411
ratio
1. the result of dividing one measured quantity by another*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2412
raw requirement
1. an environmental or customer requirement that has not been analyzed and formulated as a well-formed requirement*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.13

3.2413
RBS
1. risk breakdown structure*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** resource breakdown structure

3.2414
RDA
1. Remote Database Access*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2415
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

3.2416
reachability graph
1. a 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

3.2417
reachability set
1. the 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

3.2418
reachable marking
1. a 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

3.2419
reactivation
1. cloning a cluster following its deactivation*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.26

3.2420
read
1. to access data from a storage device or data medium. ***2.*** a data movement type that moves a data group from persistent storage within reach of the functional process which requires it*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.21

cf. destructive read, nondestructive read, write

NOTE A read is considered to include certain associated data manipulations necessary to achieve the read.

3.2421
readability
1. the ease with which a system's source code can be read and understood, especially at the detailed, statement level

3.2422
reader note
1. a comment made by a reader about a diagram and placed on the diagram page*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.97

NOTE A reader note is not part of the diagram itself, but rather is used for communication about a diagram during model development.

3.2423
reading reference
1. data storage entity or record, or interface record from another software or system containing data retrieved in a BFC*. ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.8

NOTE The number of reading references is equal to 0 for all BFC types where it is applicable.

3.2424
read-only
1. a property that causes no state changes*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.155

NOTE That is, it does no updates.

3.2425
real address
1. the address of a storage location in the main storage part of a virtual storage system

cf. virtual address

3.2426
real storage
1. the main storage portion of a virtual storage system

cf. virtual storage

3.2427
real type
1. a 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

3.2428
realization
1. the representation of interface responsibilities through specified algorithms and representation properties*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.156

NOTE 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 may 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.

3.2429
real-time
1. a 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. *Syn:* realtime, real time

3.2430
real-time scheduling theory
1. a theory for priority-based scheduling of concurrent tasks with hard deadlines

NOTE It addresses how to determine whether a group of tasks, whose individual CPU utilization is known, will meet their deadlines.

3.2431
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.41

EXAMPLE printer, filing cabinet, file folder and sheet of paper

3.2432
recommendation
1. provision that conveys advice or guidance*. ISO/IEC 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998.*3.9

NOTE: [ISO/IEC Guide 2:2004]

3.2433
record
1. a set of related data items treated as a unit. ***2.*** document stating results achieved or providing evidence of activities performed*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.9

EXAMPLE in stock control, the data for each invoice could constitute one record

3.2434
record element type (RET)
1. a user-recognizable subgroup of data elements within an ILF or EIF*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6

3.2435
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*

3.2436
recoverability
1. the degree to which object state changes resulting from failed transactions are cancelled*. ISO/IEC 10746-3:1996 Information technology — Open Distributed Processing — Reference Model: Architecture.*13.7.1.4

3.2437
recovery
1. the 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*8.1.25

cf. backward recovery, checkpoint, forward recovery

3.2438
recto
1. page on the same side (i.e. right or left) as the front cover*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.42

3.2439
RECUP
1. Repair/Enhancement/ Conversion/User support/Prevention*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

cf. maintenance (support) rate

3.2440
recursion
1. a process in which a software module calls itself. ***2.*** the process of defining or generating a process or data structure in terms of itself

cf. simultaneous recursion

3.2441
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

3.2442
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

3.2443
reengineering
1. the examination and alteration of software to reconstitute it in a new form, including the subsequent implementation of the new form. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*6.4.2. ***2.*** the complete cycle of performing reverse engineering followed by forward engineering

3.2444
reentrant
1. pertaining to a software module that can be entered as part of one process while also in execution as part of another process and still achieve the desired results. *Syn:* reenterable, re-entrant

3.2445
reentry point
1. the place in a software module at which the module is reentered following a call to another module. *Syn:* re-entry point

3.2446
reference expression
1. an expression that uniquely identifies a box, a node or function, a diagram, or a model page within an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.98

3.2447
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

3.2448
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.42

3.2449
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

3.2450
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

3.2451
referential integrity
1. a guarantee that a reference refers to an object that exists. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.157. ***2.*** a guarantee that all specified conditions for a relationship hold true. *IEEE Std 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

3.2452
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. *Syn:* recursive

3.2453
reflexive ancestor (of a class)
1. the class itself or any of its generic ancestors. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.158

cf. generic ancestor

3.2454
regression test
1. retesting to detect faults introduced by modification

3.2455
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 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.11. ***3.*** functional testing that follows modification and maintenance

3.2456
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) — Fourth Edition*

3.2457
relation
1. a set of relationships of the same relationship type. *ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*3.2.3

3.2458
relationship
1. a real-world association among one or more entities. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2. ***2.*** an association between two (not necessarily distinct) classes that is deemed relevant within a particular scope and purpose. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.159. ***3.*** a semantic connection between model elements. ***4.*** an association of interest between two entities. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***5.*** a predicate involving two or more roles with assigned values. *ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function*.3.2.1

NOTE The association is named for the sense in which the instances are related. A relationship can be represented as a time-varying binary relation between the instances of the current extents of two state classes.

3.2459
relationship instance
1. an association of specific instances of the related classes*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.160

3.2460
relationship name
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.161

NOTE [key style]

3.2461
relationship type
1. a 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

3.2462
relative address
1. an 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

3.2463
relative chain frequency
1. the 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

3.2464
release
1. a delivered version of an application which may include all or part of an application. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***2.*** collection of new and/or changed configuration items which are tested and introduced into the live environment together*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.10. ***3.*** a software version that is made formally available to a wider community. ***4.*** particular version of a configuration item that is made available for a specific purpose*. IEEE Std 828-2005 IEEE Standard for Software Configuration Management Plans.*2.12; *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.35. ***5.*** the formal notification and distribution of an approved version*. IEEE Std 828-2005 IEEE Standard for Software Configuration Management Plans.*2.1.2

EXAMPLE a test release

NOTE Release management includes defining acceptable quality levels for release, authority to authorize the release, and release procedures.

3.2465
release engineer
1. the person responsible for coordinating development toward a release

NOTE The release engineer will monitor pending issues for a given release, oversee the code freeze, and tag the release once it gets out the door.

3.2466
relevant stakeholder
1. stakeholder that is identified for involvement in specified activities and is included in a plan

3.2467
reliability
1. the ability of a system or component to perform its required functions under stated conditions for a specified period of time. ***2.*** capability of the software product to maintain a specified level of performance when used under specified conditions. *ISO/IEC 9126-1:2001, Software engineering — Product quality — Part 1: Quality model.* 6.2.

cf. availability, MTBF

NOTE Wear or aging does not occur in software. Limitations in reliability are due to faults in requirements, design, and implementation. Failures due to these faults depend on the way the software product is used and the program options selected rather than on elapsed time.

3.2468
reliability growth
1. the improvement in reliability that results from correction of faults

3.2469
relocatable
1. pertaining to code that can be loaded into any part of main memory

cf. relocating loader

NOTE The starting address is established by the loader, which then adjusts the addresses in the code to reflect the storage locations into which the code has been loaded.

3.2470
relocatable address
1. an address that is to be adjusted by the loader when the computer program containing the address is loaded into memory

cf. absolute address

3.2471
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

3.2472
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

3.2473
relocating assembler
1. an assembler that produces relocatable code

cf. absolute assembler

3.2474
relocating loader
1. a 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. *Syn:* relative loader

cf. absolute loader

3.2475
relocation dictionary
1. the part of an object module or load module that identifies the addresses that must be adjusted when a relocation occurs

3.2476
relocation transparency
1. a distribution transparency which masks relocation of an interface from other interfaces bound to it. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.5

3.2477
relocator
1. an object providing the relocation function. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*14.3.1.1

3.2478
remote job entry (RJE)
1. submission of jobs through a remote input device connected to a computer through a data link. *Syn:* remote batch entry

3.2479
remote terminal emulator (RTE)
1. a 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

3.2480
repair
1. the correction of defects that have resulted from errors in external design, internal design, or code. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual. Syn:* defect removal

EXAMPLE missing functions that do not result in application failure (external design error) or errors resulting in a stop-run situation (code error)

3.2481
reparent
1. to move the changes developed under one branch into another

NOTE The changes are not committed to the original branch.

3.2482
repeatability (of results of measurements)
1. closeness of the agreement between the results of successive measurements of the same measurand carried out under the same conditions of measurement. *ISO/IEC TR 14143-3:2003, Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods.*3.8

NOTE These conditions are called repeatability conditions. Repeatability conditions include the same measurement procedure, the same observer, the same measuring instrument, used under the same conditions; the same location; repetition over a short period of time. Repeatability may be expressed quantitatively in terms of the dispersion characteristics of the results. [International vocabulary of basic and general terms in metrology, 1993, definition 3.6]

3.2483
repetitive addressing
1. a 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

3.2484
replication
1. copying a software product from one medium to another*. ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.13

3.2485
replication schema
1. a 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*16.7.1.1

3.2486
replication transparency
1. a distribution transparency which masks the use of a group of mutually behaviorally compatible objects to support an interface*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.6

NOTE Replication is often used to enhance performance and availability.

3.2487
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 A repo master can reduce repo bloat through repo surgery.

3.2488
repo master
1. the person in charge of the version control system's repository

NOTE one who has rights to perform repo surgery

3.2489
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 Through repo surgery, a repo master can perform operations that the version control system does not directly support.

3.2490
report
1. 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 The output medium used is not relevant for FPA and it can pertain to both an external output and an external inquiry.

3.2491
report performance
1. [Process] the process of collecting and distributing performance information, including status reports, progress measurements, and forecasts*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2492
report standard
1. a standard that describes the characteristics of describing results of engineering and management activities

3.2493
repository
1. a collection of all software-related artifacts belonging to a system. ***2.*** the location/format in which such a collection is stored

NOTE Artifacts are, for example, the software engineering environment.

3.2494
representation
1. a likeness, picture, drawing, block diagram, description, or symbol that logically portrays a physical, operational, or conceptual image or situation*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.14. ***2.*** one or more properties used by an algorithm for the realization of a responsibility*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.162

3.2495
representation property
1. a property on which an algorithm operates*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.163

3.2496
representation standard
1. a standard that describes the characteristics of portraying aspects of an engineering or management product

3.2497
reproducibility (of results of measurements)
1. closeness of the agreement between the results of measurements of the same measurand carried out under changed conditions of measurement*. ISO/IEC TR 14143-3:2003, Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods.*3.9

NOTE A valid statement of reproducibility requires specification of the conditions changed. The changed conditions may include the principle of measurement; method of measurement; observer; measuring instrument; reference standard; location; conditions of use; time. Reproducibility may be expressed quantitatively in terms of the dispersion characteristics of the results. Results are here usually understood to be corrected results. [International vocabulary of basic and general terms in metrology, 1993, definition 3.7]

3.2498
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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.164. ***2.*** a call issued by a client that causes a service to be performed*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.12

cf. message

NOTE Specifically, a request may 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.

3.2499
request for change
1. form or screen used to record details of a request for a change to any configuration item within a service or infrastructure*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.11

3.2500
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) — Fourth Edition*

3.2501
request for proposal (RFP)
1. a document used by the acquirer as a means to announce intention to potential bidders to acquire a specified system, product, or service*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.24. ***2.*** a request for services, research, or a product prepared by a customer and delivered to prospective developers with the expectation that prospective developers will respond with their proposed cost, schedule, and development approach*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2. ***3.*** a type of procurement document used to request proposals from prospective sellers of products or services. In some application areas, it may have a narrower or more specific meaning*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***4.*** 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. *Syn:* request for tender, solicitation package

3.2502
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) — Fourth Edition*

3.2503
requested change
1. [Output/Input] 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) — Fourth Edition*

3.2504
required inputs
1. set of items necessary to perform the minimum testing tasks mandated within a life cycle activity*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.27

3.2505
required outputs
1. set of items produced as a result of performing the minimum testing tasks mandated within any life cycle activity*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.28

3.2506
requirement
1. a condition or capability needed by a user to solve a problem or achieve an objective. ***2.*** a 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 ***3.*** a documented representation of a condition or capability as in (1) or (2) ***4.*** a condition or capability that must be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed document. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. design requirement, functional requirement, implementation requirement, interface requirement, performance requirement, physical requirement

3.2507
requirement standard
1. a standard that describes the characteristics of a requirements specification

3.2508
requirements allocation
1. the assignment or budgeting of top-level functional or nonfunctional requirements among the lower-level partitioned functions for accomplishment

NOTE In this manner, the system elements that perform all or part of specific requirements are identified.

3.2509
requirements analysis
1. the process of studying user needs to arrive at a definition of system, hardware, or software requirements. ***2.*** the process of studying and refining system, hardware, or software requirements. ***3.*** a systematic investigation of user requirements to arrive at a definition of a system. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.04. ***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

3.2510
requirements derivation
1. the 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

3.2511
requirements document
1. document containing any combination of requirements or regulations to be met by a COTS software product. *ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.6

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

3.2512
requirements elicitation
1. the process through which the acquirers (customers or users) and the suppliers (contractor) of a system discover, review, articulate, understand, and document the users' needs and the constraints on the system and the development activity ***2.*** the use of systematic techniques, such as prototypes and structured surveys, to proactively identify and document customer and end-user needs

3.2513
requirements engineering
1. the science and discipline concerned with analyzing and documenting requirements

NOTE It comprises needs analysis, requirements analysis, and requirements specification.

3.2514
requirements flow-down
1. the systematic decomposition of system requirements into allocated and derived requirements, appropriately assigned to low-level functional components

3.2515
requirements partitioning
1. the separation or decomposing of a top-level requirement or design into successively lower-level detailed requirements or design. *Syn:* decomposition

3.2516
requirements phase
1. the period of time in the software life cycle during which the requirements for a software product are defined and documented

3.2517
requirements review
1. a 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 Types include system requirements review, software requirements review.

3.2518
requirements specification
1. a document that specifies the requirements for a system or component

cf. design description, functional specification, performance specification

NOTE Typically included are functional requirements, performance requirements, interface requirements, design requirements, and development standards.

3.2519
requirements specification language
1. a specification language with special constructs and, sometimes, verification protocols, used to develop, analyze, and document hardware or software requirements

cf. design language

3.2520
requirements traceability
1. the identification and documentation of the derivation path (upward) and allocation/ flow-down path (downward) of requirements in the requirements hierarchy ***2.*** discernable association between a requirement and related requirements, implementations, and verifications

3.2521
requirements traceability matrix (RTM)
1. a table that links requirements to their origin and traces them throughout the project life cycle*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2522
requirements traceability tool
1. a software development tool that establishes a traceability among itemized software requirements specifications, design elements, code elements, and test cases

NOTE It also supports various associated query, analysis, and report-generation capabilities.

3.2523
reserve
1. a provision in the project management plan to mitigate cost and/or schedule risk. Often used with a modifier (e.g., management reserve, contingency reserve) to provide further detail on what types of risk are meant to be mitigated*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* buffer, contingency allowance+H47

3.2524
reserve analysis
1. [Technique] 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) — Fourth Edition*

3.2525
reserved word
1. a 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

3.2526
reset
1. to set a variable, register, or other storage location back to a prescribed state

cf. clear, initialize

3.2527
residual control
1. a 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

3.2528
residual risk
1. a risk that remains after risk responses have been implemented. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2529
resource
1. 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) — Fourth Edition.* ***2.*** asset that is utilized or consumed during the execution of a process. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.37; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.25. ***3.*** a role (with respect to that action) in which the enterprise object fulfilling the role is essential to the action, requires allocation, or may become unavailable. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.3.3. ***4.*** an enterprise object which is essential to some behavior and which requires allocation or may become unavailable. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.3.3. ***5.*** people, procedure, software, information, equipment, consumable, infrastructure, capital and operating funds, and time. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.13

EXAMPLE diverse entities such as funding, personnel, facilities, capital equipment, tools, and utilities such as power, water, fuel and communication infrastructures

NOTE Allocation of a resource may constrain other behaviors for which that resource is essential. Resources may be reusable, renewable or consumable. A consumable resource may 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).

3.2530
resource breakdown structure (RBS)
1. a hierarchical structure of resources by resource category and resource type used in resource leveling schedules and to develop resource-limited schedules, and which may be used to identify and analyze project human resource assignments. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2531
resource calendar
1. a calendar of working days and nonworking days that determines those dates on which each specific resource is idle or can be active. Typically defines resource specific holidays and resource availability periods. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. project calendar

3.2532
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) — Fourth Edition*

3.2533
resource leveling
1. [Technique] any form of schedule network analysis in which scheduling decisions (start and finish dates) are driven by resource constraints (e.g., limited resource availability or difficult-to-manage changes in resource availability levels). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2534
resource management
1. the identification, estimation, allocation, and monitoring of the means used to develop a product or perform a service

EXAMPLE estimating

3.2535
resource monitor task
1. a task ensuring sequential access to a resource

3.2536
resource-limited schedule
1. a project schedule whose schedule activity, scheduled start dates and scheduled finish dates reflect expected resource availability

3.2537
respecialize
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.165

cf. specialize, unspecialize

3.2538
response time
1. the 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

3.2539
responsibility
1. a generalization of properties (attributes, participant properties, and operations) and constraints. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.166

NOTE 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 may apply to each instance of the class (instance-level) or to the class as a whole (class-level).

3.2540
responsibility assignment matrix (RAM)
1. [Tool] a structure that relates the project organizational breakdown structure to the work breakdown structure to help ensure that each component of the project's scope of work is assigned to a person or team. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2541
restart
1. to cause a computer program to resume execution after a failure, using status and results recorded at a checkpoint

3.2542
restart point
1. a point in a computer program at which execution can be restarted following a failure. *Syn:* rescue point

3.2543
result
1. the information returned to the client. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.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) — Fourth Edition*

cf. product, deliverable

NOTE may include values as well as status information indicating that exceptional conditions were raised in attempting to perform the requested service

3.2544
retainage
1. portion of a contract payment that is withheld until contract completion to ensure full performance of the contract terms

3.2545
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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes*.4.38; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes*.4.26. ***2.*** removal of support from an operational system or component. ***3.*** permanent removal of a system or component from its operational environment

3.2546
retirement phase
1. the period of time in the software life cycle during which support for a software product is terminated

cf. software life cycle, system life cycle

3.2547
retrospective trace
1. a trace produced from historical data recorded during the execution of a computer program

cf. execution trace, subroutine trace, symbolic trace, variable trace

NOTE This differs from an ordinary trace, which is produced cumulatively during program execution.

3.2548
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.*** a computer instruction or process that performs the transfer in (1)

cf. return code

3.2549
return code
1. a code used to influence the execution of a calling module following a return from a called module

3.2550
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

3.2551
return value
1. a value assigned to a parameter by a called module for access by the calling module

3.2552
reusability
1. the degree to which an asset can be used in more than one software system, or in building other assets. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.15. ***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 Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.15

cf. generality

3.2553
reusability GSC
1. one of the 14 general system characteristics describing the degree to which the application and the code in the application have been specifically designed, developed, and supported to be usable in other applications. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2554
reusable
1. pertaining to a software module or other work product that can be used in more than one computer program or software system

3.2555
reusable product
1. 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 Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.29

EXAMPLE commercial off-the-shelf (COTS) products, acquirer-furnished products, products in reuse libraries, and preexisting developer products

NOTE Each use may include all or part of the product and may 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.

3.2556
reusable software product
1. a software 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 Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.25

EXAMPLE COTS software products, acquirer-furnished software products, software products in reuse libraries, and pre-existing developer software products

NOTE Each use may include all or part of the software product and may involve its modification. This term can be applied to any software product (for example, requirements, architectures), not just to software itself

3.2557
reuse
1. the use of an asset in the solution of different problems. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.16. ***2.*** building a software system at least partly from existing pieces to perform a new application

3.2558
reuse sponsor
1. a member of the organization's management who authorizes, approves, promotes, and obtains the funding and other resources for the Reuse Program. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.17

3.2559
reused source statement
1. unmodified source statement obtained for the product from an external source

3.2560
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

3.2561
reversible execution
1. a 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. *Syn:* backward execution, playback, replay, reverse execution

3.2562
review
1. a 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 ***2.*** a 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 Std 1028-2008 IEEE Standard for Software Reviews and Audits.*3.5

3.2563
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) — Fourth Edition*

3.2564
RFI
1. request for information. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2565
RFP
1. request for proposal. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2566
RFQ
1. request for quotation. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2567
risk
1. an uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** the 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 Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.30. ***3.*** the combination of the probability of an event and its consequence*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.5; *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.14. ***4.*** a measure that combines both the likelihood that a system hazard will cause an accident and the severity of that accident*. IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.3. ***5.*** a function of the probability of occurrence of a given threat and the potential adverse consequences of that threat's occurrence*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.12. ***6.*** the combination of the probability of occurrence and the consequences of a given future undesirable event*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.30

NOTE Generally used only when there is at least the possibility of negative consequences. In some situations, risk arises from the possibility of deviation from the expected outcome or event. See ISO/IEC Guide 51 for issues related to safety. Risk can be associated with software, systems, products, and projects.

3.2568
risk acceptance
1. acknowledgment of a risk factor's existence along with a decision to accept the consequences if the corresponding problem occurs ***2.*** [Technique] a risk response planning technique that indicates that the project team has decided not to change the project management plan to deal with a risk, or is unable to identify any other suitable response strategy*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***3.*** the decision to accept a risk*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.6. *Syn:* risk assumption

NOTE Risk acceptance depends on risk criteria.

3.2569
risk action request
1. the recommended treatment alternatives and supporting information for one or more risks determined to be above a risk threshold*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.7

3.2570
risk analysis
1. the process of examining identified risk factors for probability of occurrence, potential loss, and potential risk-handling strategies

3.2571
risk avoidance
1. a course of action that removes a risk factor from further consideration. ***2.*** [Technique] a risk response planning technique for a threat that creates changes to the project management plan that are meant to either eliminate the risk or to protect the project objectives from its impact*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

EXAMPLE changing the requirements, extending the schedule, or transferring the risk factor to another domain

3.2572
risk breakdown structure (RBS)
1. [Tool] a hierarchically organized depiction of the identified project risks arranged by risk category and subcategory that identifies the various areas and causes of potential risks. The risk breakdown structure is often tailored to specific project types*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2573
risk category
1. a class or type of risk*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.8 ***2.*** a group of potential causes of risk. Risk causes may be grouped into categories such as technical, external, organizational, environmental, or project management. A category may include subcategories such as technical maturity, weather, or aggressive estimating*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. source

EXAMPLE technical, legal, organizational, safety, economic, engineering, cost, schedule

NOTE A risk category is a characterization of a source of risk.

3.2574
risk criteria
1. the terms of reference by which the significance of risk is assessed*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.9

NOTE 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.

3.2575
risk dimension
1. a perspective from which risk assessment is being made for the system*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.13

EXAMPLE safety, economic, security

3.2576
risk exposure
1. the potential loss presented to an individual, project, or organization by a risk*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.10. ***2.*** a function of the likelihood that the risk will occur and the magnitude of the consequences of its occurrence*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.10. ***3.*** the product of probability times potential loss for a risk factor

NOTE Risk exposure is commonly defined as the product of a probability and the magnitude of a consequence, that is, an expected value or expected exposure.

3.2577
risk factor
1. a potential problem that would be detrimental to a planned activity, project, or program, characterized by the probability of problem occurrence (0 _ p _ 1) and a potential loss (of life, money, property, reputation, and so on) should the problem occur

NOTE Both probability and potential loss might change over time.

3.2578
risk handling
1. a course of action taken in response to a risk factor

NOTE includes risk acceptance, risk avoidance, risk transfer, and risk mitigation

3.2579
risk identification
1. an organized, systematic approach to determining the risk factors associated with a planned activity, project, or program

cf. identify risks

3.2580
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 Larger rlfs indicate better mitigation strategies.

3.2581
risk management
1. an 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 ***3.*** coordinated activities to direct and control an organization with regard to risk. *ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.15

NOTE The primary goal of risk management is to identify and respond to potential problems with sufficient lead-time to avoid a crisis situation. Includes initial identification and handling of risk factors as well as continuous risk management.

3.2582
risk management plan
1. a description of how the elements and resources of the risk management process will be implemented within an organization or project. *ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.11. ***2.*** [Output/Input]. The document describing how project risk management will be structured and performed on the project. It is contained in or is a subsidiary plan of the project management plan. Information in the risk management plan varies by application area and project size. The risk management plan is different from the risk register that contains the list of project risks, the results of risk analysis, and the risk responses. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2583
risk management process
1. a continuous process for systematically identifying, analyzing, treating, and monitoring risk throughout the life cycle of a product or service. *ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.12

3.2584
risk management system
1. set of elements of an organization's management system concerned with managing risk. *ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.13

NOTE 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. [ISO Guide 73:2002]

3.2585
risk metric
1. an objective measure associated with a risk factor to be mitigated

3.2586
risk mitigation
1. a course of action taken to reduce the probability of and potential loss from a risk factor. ***2.*** [Technique] a risk response planning technique associated with threats that seeks to reduce the probability of occurrence or impact of a risk to below an acceptable threshold. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

NOTE includes executing contingency plans when a risk metric crosses a predetermined threshold (when a risk factor becomes a problem)

3.2587
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

3.2588
risk profile
1. a chronological record of a risk's current and historical risk state information*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.14

3.2589
risk reduction
1. reducing the probability and/or potential impact of a risk factor

NOTE Risk reduction might involve research, prototyping, and other means of exploration.

3.2590
risk register
1. [Output/Input] the document containing the results of the qualitative risk analysis, quantitative risk analysis, and risk response planning. The risk register details all identified risks, including description, category, cause, probability of occurring, impact(s) on objectives, proposed responses, owners, and current status*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2591
risk state
1. the current project risk information relating to an individual risk*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.15

NOTE The information concerning an individual risk may include 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.

3.2592
risk threshold
1. a condition that triggers some stakeholder action*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.16

NOTE Different risk thresholds may be defined for each risk, risk category or combination of risks based on differing risk criteria.

3.2593
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) — Fourth Edition*

3.2594
risk transfer
1. transferring responsibility for managing a risk factor to another organization or functional entity better able to mitigate the risk factor

3.2595
risk transference
1. [Technique] a risk response planning technique that 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) — Fourth Edition*

3.2596
risk treatment
1. the process of selection and implementation of measures to modify risk. *ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering--Life cycle processes--Risk management.*3.17. ***2.*** means of modifying risk

NOTE Risk treatment measures can include avoiding, optimizing, transferring, or retaining risk.

3.2597
risk trigger
1. the predetermined threshold value of a risk metric that triggers invocation of a contingency plan when the risk metric crosses the threshold

3.2598
RJE
1. remote job entry

3.2599
RM-ODP
1. Reference Model of Open Distributed Processing. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview* ***2.*** Open Distributed Processing: Reference Model. *ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*4

3.2600
robotics
1. the techniques involved in designing, building, and using robots. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.13

3.2601
robustness
1. the 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

3.2602
ROI
1. return on investment

3.2603
role
1. the 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) — Fourth Edition.* ***3.*** the 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

NOTE Each instance of a role has a minimum and maximum cardinality, and may be attributed. The direction of the role indicates how to read the name of the role.

3.2604
role name
1. a name that more specifically names the nature of a related value class or state class. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.167. ***2.*** a name assigned to a foreign key attribute to represent the use of the foreign key in

the entity. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.167

NOTE For a relationship, a role name is a name given to a class in a relationship to clarify the participation of that class in the relationship, that is, connote the role played by a related instance. For an attribute, a role name is a name used to clarify the sense of the value class in the context of the class for which it is a property.

3.2605
roll in
1. to transfer data or computer program segments from auxiliary storage to main storage

cf. roll out, swap

3.2606
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

3.2607
rolling wave planning
1. [Technique] a form of progressive elaboration planning where the work to be accomplished in the near term is planned in detail at a low level of the work breakdown structure, while the work far in the future is planned at a relatively high level of the work breakdown structure, but the detailed planning of the work to be performed within another one or two periods in the near future is done as work is being completed during the current period. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2608
root arrow segment
1. the arrow segment of a junction from which other arrow segments branch or to which other arrow segments join. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.99. *Syn:* root, root segment

3.2609
root cause
1. source of a defect such that if it is removed, the defect is decreased or removed

3.2610
root cause analysis
1. [Technique] 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) — Fourth Edition. Syn:* root-cause analysis

3.2611
root compiler
1. a compiler whose output is a machine independent, intermediate-level representation of a program

NOTE A root compiler, when combined with a code generator, comprises a full compiler.

3.2612
root-cause analysis
1. determination of a potential problem's (a risk factor's) underlying cause or causes. *Syn*: root cause analysis.

3.2613
routine
1. a subprogram that is called by other programs and subprograms. ***2.*** a function or procedure invocable for a single purpose. ***3.*** a program, or part of a program, that may have some general or frequent use*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.05.04

cf. coroutine, subroutine

NOTE The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages.

3.2614
RPC
1. Remote Procedure Call*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.2615
RTM
1. requirements traceability matrix*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.2616
rule
1. a single column through the condition and action entry parts of the decision table, defining a unique set of conditions to be satisfied and the actions to be taken in consequence*. ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.4

NOTE A rule is satisfied if all conditions meet the condition entries of the rule.

3.2617
Rule and Constraint Language (RCL)
1. a declarative specification language that is used to express the realization of responsibilities and to state queries*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.168

cf. specification language

3.2618
rule-based language
1. a 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

3.2619
run
1. in software engineering, a single, usually continuous, execution of a computer program. ***2.*** to execute a computer program

cf. run time

3.2620
run time
1. the instant at which a computer program begins to execute. ***2.*** the period of time during which a computer program is executing

cf. execution time

3.2621
RUR
1. Reference User Requirements. *ISO/IEC TR 14143-4:2002, Information technology — Software measurement — Functional size measurement — Part 4: Reference model.*4

3.2622
safety
1. the expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.14

3.2623
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 Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.4

3.2624
SAM
1. software asset management

3.2625
SAM owner
1. individual at a senior organization-wide level who is identified as being responsible for SAM. *ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 11

3.2626
sample instance diagram
1. a form of presenting example instances in which instances are shown as separate graphic objects. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.169

cf. sample instance table

NOTE The graphic presentation of instances can be useful when only a few instances are presented.

3.2627
sample instance table
1. a form of presenting example instances in which instances are shown as a tabular presentation. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.170

cf. sample instance diagram

NOTE The tabular presentation of instances can be useful when several instances of one class are to be presented.

3.2628
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

3.2629
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

3.2630
scalar
1. a value that is atomic*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.171

cf. collection-valued

NOTE that is, having no parts

3.2631
scalar-valued class
1. a class in which each instance is a single value*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.173

cf. collection-valued class

3.2632
scalar-valued property
1. a property that maps to a scalar-valued class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.174. *Syn:* scalar property

cf. collection-valued property

3.2633
scale
1. a set of values with defined properties*. ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview.*4.27. ***2.*** ordered set of values, continuous or discrete, or a set of categories to which the attribute is mapped*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.35; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.48

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 point but also possess an absolute zero

NOTE The type of scale depends on the nature of the relationship between values on the scale. Metrics using nominal or ordinal scales produce qualitative data, and metrics using interval and ratio scales produce quantitative data.

3.2634
SCCS
1. Server Conversion Code Sets*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.2635
scenario
1. a step-by-step description of a series of events that may occur concurrently or sequentially*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2. ***2.*** an account or synopsis of a projected course of events or actions*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2. ***3.*** a description of a specific sequence of actions. *Syn:* script, set, suite

cf. use case

3.2636
schedule baseline
1. a specific version of the schedule model used to compare actual results to the plan to determine if preventive or corrective action is needed to meet the project objectives*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2637
schedule compression
1. [Technique] shortening the project schedule duration without reducing the project scope*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. crashing, fast tracking

3.2638
schedule development
1. process of creating the project schedule by analyzing activity sequences, activity durations, resource requirements, and schedule constraints

3.2639
schedule management plan
1. [Output/Input] the document that establishes criteria and the activities for developing and controlling the project schedule. It is contained in, or is a subsidiary plan of, the project management plan*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2640
schedule model
1. [Tool] a model used in conjunction with manual methods or project management software to perform schedule network analysis to generate the project schedule for use in managing the execution of a project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* project schedule

3.2641
schedule network analysis
1. [Technique] 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) — Fourth Edition. Syn:* schedule analysis

cf. critical path method, critical chain method, resource leveling

3.2642
schedule performance index (SPI)
1. a measure of schedule efficiency on a project. It is the ratio of earned value (EV) to planned value (PV). The SPI = EV divided by PV*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2643
schedule variance (SV)
1. a measure of schedule performance on a project. It is the algebraic difference between the earned value (EV) and the planned value (PV). SV = EV minus PV*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2644
scheduled finish date (SF)
1. the point in time that work was scheduled to finish on a schedule activity. The scheduled finish date is normally within the range of dates delimited by the early finish date and the late finish date. It may reflect resource leveling of scarce resources*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* planned finish date (PF)

3.2645
scheduled start date (SS)
1. the point in time that work was scheduled to start on a schedule activity. The scheduled start date is normally within the range of dates delimited by the early start date and the late start date. It may reflect resource leveling of scarce resources*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* planned start date (PS)

3.2646
scheduler
1. a computer program, usually part of an operating system, that schedules, initiates, and terminates jobs

3.2647
SCI
1. Software Configuration Item. *ISO/IEC TR 15846:1998, Information technology — Software life cycle processes — Configuration Management.*5.1

3.2648
SCM
1. Software Configuration Management. *ISO/IEC TR 15846:1998, Information technology — Software life cycle processes — Configuration Management.*5.1

3.2649
SCMPI
1. Software Configuration Management Planned Information. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2650
SCN
1. specification change notice

3.2651
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) — Fourth Edition.* ***2.*** the behavior that system is expected to exhibit

cf. project scope, product scope

3.2652
scope baseline
1. an approved specific version of the detailed scope statement, work breakdown structure (WBS), and its associated WBS dictionary. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2653
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) — Fourth Edition*

3.2654
scope creep
1. additional functionality that was not specified in the original requirements but is identified as the requirements and scope are clarified and the functions are defined. *ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10. ***2.*** adding features and functionality (project scope) without addressing the effects on time, costs, and resources, or without customer approval. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***3.*** additional functionality that was not specified in the original requirements, but is identified as the scope is being clarified and the functions defined. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual. Syn:* scope gallop

3.2655
scope management plan
1. [Output/Input] the document that describes how the project scope will be defined, developed, and verified and how the work breakdown structure will be created and defined, and that provides guidance on how the project scope will be managed and controlled by the project management team. It is contained in or is a subsidiary plan of the project management plan. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2656
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 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.16

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 The Scope of the FSM is determined by the purpose for measuring the software.

3.2657
screen dump
1. representation of what the user will see while using the software*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.43

3.2658
S-curve
1. graphic display of cumulative costs, labor hours, percentage of work, or other quantities, plotted against time. Used to depict planned value, earned value, and actual cost of project work. The name derives from the S-like shape of the curve (flatter at the beginning and end, steeper in the middle) produced on a project that starts slowly, accelerates, and then tails off. Also a term used to express the cumulative likelihood distribution that is a result of a simulation, a tool of quantitative risk analysis*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2659
SDD
1. software design description. ***2.*** software design document

3.2660
SDP
1. software development plan

3.2661
SDR
1. system design review

3.2662
second normal form
1. result of a normalization process that transforms groups of data so that each non-key attribute depends on the key attribute(s) of the group of data and all parts of the key attribute(s). *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.2663
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) — Fourth Edition*

3.2664
secondary window
1. window that contains information that depends on information in another window (the primary window)*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.43

3.2665
security
1. the protection of system items from accidental or malicious access, use, modification, destruction, or disclosure*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.15.

2. 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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.39. ***3.*** all aspects related to defining, achieving, and maintaining confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of a system*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.27

NOTE Security also pertains to personnel, data, communications, and the physical protection of computer installations.

3.2666
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*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*3.1. ***2.*** An independent accreditation body's certification that an IT system meets a predetermined security standard*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*3.1

3.2667
security authority
1. the administrator responsible for the implementation of a security policy*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*15.1.2

3.2668
security branch
1. a branch, created at the time of a release, to which only security commits are made

3.2669
security domain
1. a domain in which the members are obliged to follow a security policy established and administered by a security authority*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*15.1.3

NOTE The security authority is the controlling object for the security domain.

3.2670
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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*15.1.4

3.2671
security kernel
1. a 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

3.2672
security policy
1. the rules for need-to-know and access-to-information at each project organization level. ***2.*** a set of rules that constrains one or more sets of activities of one or more sets of objects*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*15.1.1

3.2673
SEE
1. Software Engineering Environment*. ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5

3.2674
SEE service
1. one or more service operations to support life cycle activities for the SEE*. ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.2

NOTE A SEE Service supplier provides a SEE Service for a SEE Service acquirer.

3.2675
SEE service acquirer
1. an actor that acquires an SEE service*. ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.6

3.2676
SEE service supplier
1. an actor that supplies an SEE service*. ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.7

3.2677
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.*** a 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)

EXAMPLE the accounts payable segment of a financial system

3.2678
selective dump
1. a dump of designated storage location areas only

cf. change dump, dynamic dump, memory dump, postmortem dump, snapshot dump, static dump

3.2679
selective trace
1. a variable trace that involves only selected variables

cf. execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

3.2680
self-descriptiveness
1. the 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

3.2681
self-documented
1. pertaining to source code that contains comments explaining its objectives, operation, and other information useful in understanding and maintaining the code

3.2682
self-relative address
1. an 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

3.2683
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) — Fourth Edition*

3.2684
semantic agreement
1. a passive interconnection in which two things agree on a common interpretation of statements (symbol arrangements) by reference to a shared thing or phenomenon*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.12

cf. syntactic agreement

3.2685
semantic error
1. an error resulting from a misunderstanding of the relationship of symbols or groups of symbols to their meanings in a given language

cf. syntactic error

3.2686
semantics
1. the meaning of the syntactic components of a language*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.175. ***2.*** the relationships of symbols or groups of symbols to their meanings in a given language

cf. syntax

3.2687
semaphore
1. a shared variable used to synchronize concurrent processes by indicating whether an action has been completed or an event has occurred

cf. flag, indicator

3.2688
SEMDM
1. software engineering metamodel for development methodologies*. ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies.*4.2

3.2689
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) — Fourth Edition*

3.2690
separate documentation
1. documentation that may be used independently of the software

cf. embedded documentation

EXAMPLE printed manual and freestanding hypertext system

3.2691
sequence activities
1. [Process] the process of identifying and documenting relationships among the project activities*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2692
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. *Syn:* serial (2)

cf. consecutive

3.2693
sequential clustering
1. a task-structuring criterion in which objects that are constrained to execute sequentially are mapped to a task

3.2694
sequential cohesion
1. a 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

3.2695
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

3.2696
serial construct
1. a program construct consisting of a sequence of steps not involving a decision or loop. *Syn:* sequential construct

3.2697
server
1. a hardware system or software program which provides a service to clients. ***2.*** a process implementing one or more operations on one or more objects. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.14

3.2698
server-side
1. a 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

3.2699
service
1. performance of activities, work, or duties associated with a product. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.40; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.36

cf. product, result, deliverable

3.2700
service desk
1. customer facing support group who do a high proportion of the total support work. *ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.12

3.2701
service export
1. an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*14.5.1.2

3.2702
service import
1. an 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*14.5.1.3

3.2703
service level agreement (SLA)
1. written agreement between a service provider and a customer that documents services and agreed service levels*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.13

3.2704
service management
1. management of services to meet the business requirements*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.14

3.2705
service offer
1. information about an interface including both an identifier for the interface and its computational interface signature type*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*14.5.1.1

NOTE The identifier enables binding to the interface. The computational signature enables a trader to ensure service import selects service offers which will interact in the way expected by the importing object. Additional information in the service offer can be used to provide greater discrimination than that embodied in interface signatures.

3.2706
service primitive
1. an 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

3.2707
service provider
1. the organization aiming to achieve ISO/IEC 20000*. ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification.*2.15

3.2708
set
1. a collection class with no duplicate members and where order is irrelevant*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.176

cf. bag, list

3.2709
set-up time
1. the 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

3.2710
SF
1. start-to-finish*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2711
shadow class
1. a class presented in a view that is specified in some other view. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.177

3.2712
shell
1. a computer program or routine that provides an interface between the user and a computer system or program

3.2713
should-cost estimate
1. estimate of the cost of a product or service used to evaluate the reasonableness of a supplier's proposed price

3.2714
signal
1. an atomic shared action resulting in one-way communication from an initiating object to a responding object. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.1. ***2.*** a variation of a physical quantity used to represent data. *ISO/IEC 2382-1:1993, Information technology--Vocabulary--Part 1: Fundamental terms.*01.02.01

NOTE A signal is an interaction.

3.2715
signal interface
1. an interface in which all the interactions are signals. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.6

3.2716
signal interface signature
1. an interface signature for a signal interface. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.11

NOTE A signal interface signature comprises a finite set of action templates, one for each signal type in the interface. Each action template comprises the name for the signal, the number, names and types of its parameters, and an indication of causality (initiating or responding, but not both) with respect to the object which instantiates the template.

3.2717
signature
1. the 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 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.15. ***2.*** a 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.*** a statement of what the interface to a responsibility ""looks like.". *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.178

NOTE A signature consists of the responsibility name, along with a property operator and the number and type of its arguments, if any. A type (class) may be specified for each argument in order to limit the argument values to being instances of that class

3.2718
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.45

NOTE Information of different types may be indicated by symbols or graphics of different types.

3.2719
simple buffering
1. a buffering technique in which a buffer is allocated to a computer program for the duration of the program's execution

cf. dynamic buffering

3.2720
simple token
1. a 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

3.2721
simplicity
1. the 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

3.2722
simulation
1. a model that behaves or operates like a given system when provided a set of controlled inputs. ***2.*** the process of developing or using a model as in (1). ***3.*** the use of a data processing system to represent selected behavioral characteristics of a physical or abstract system. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.01

cf. emulation

3.2723
simulator
1. a device, computer program, or system that behaves or operates like a given system when provided a set of controlled inputs

cf. emulator

3.2724
simultaneous
1. pertaining to the occurrence of two or more events at the same instant of time

cf. concurrent

3.2725
simultaneous recursion
1. a situation in which two software modules call each other

3.2726
single-hit decision table
1. a 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

3.2727
single-level encoding
1. a microprogramming technique in which different microoperations are encoded as different values in the same field of a microinstruction. *Syn:* single level encoding

cf. two-level encoding

3.2728
single-step operation
1. a debugging technique in which a single computer instruction, or part of an instruction, is executed in response to an external signal. *Syn:* single-step execution, step-by-step operation

3.2729
single-valued property
1. a property with a single-valued mapping. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.179

cf. multi-valued property

3.2730
sizing
1. the process of estimating the amount of computer storage or the number of source lines required for a software system or component

cf. timing

3.2731
skeleton
1. the object-interface-specific ORB component which assists an object adapter in passing requests to particular methods*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.16

3.2732
SLCP
1. Software Life Cycle Processes*. ISO/IEC TR 9126-3:2003, Software engineering — Product quality — Part 3: Internal metrics.*5

3.2733
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

3.2734
slush
1. a preparation for a feature or code freeze

NOTE During this period, developers will commit code they have been working on but are discouraged from starting on new elements. If a freeze lasts for a long time, a slush might be introduced to ease its passing by allowing in some extra elements.

3.2735
SME
1. subject-matter expert*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.44

3.2736
SMIR
1. Server Makes It Right*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.2737
snapshot dump
1. a 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

3.2738
SNCS
1. Server Native Code Set*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.3

3.2739
soft copy image
1. nonpermanent output of information in audio or visual format*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.05

EXAMPLE a video display

3.2740
soft failure
1. a failure that permits continued operation of a system with partial operational capability

cf. hard failure

3.2741
software
1. all or part of the programs, procedures, rules, and associated documentation of an information processing system*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.08. ***2.*** computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.32. ***3.*** program or set of programs used to run a computer*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.46

cf. application software

EXAMPLE command files, job control language

NOTE includes firmware, documentation, data, and execution control statements

3.2742
software acquisition process
1. the period of time that begins with the decision to acquire a software product and ends when the product is no longer available for use*. IEEE Std 1062, 1998 Edition (R2002) IEEE Recommended Practice for Software Acquisition (includes IEEE Std 1062a).*3.8

NOTE The software acquisition process typically includes nine steps associated with planning the organizational strategy, implementing an organization's process, determining the software requirements, identifying potential suppliers, preparing contract requirements, evaluating proposals and selecting the supplier, managing supplier performance, accepting the software, and using the software.

3.2743
software asset management (SAM)
1. effective management, control and protection of software assets within an organization*. ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 12

3.2744
software characteristic
1. an inherent, possibly accidental, trait, quality, or property of software*. IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

EXAMPLE functionality, performance, attributes, design constraints, number of states, lines of branches

3.2745
software component (SC)
1. a general term used to refer to a software system or an element, such as module, unit, data, or document. *IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology*.2.2. ***2.*** a 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

3.2746
software configuration item (SCI)
1. a software entity that has been established as a configuration item

cf. hardware configuration item

NOTE The SCI exists where functional allocations have been made that clearly distinguish equipment functions from software functions and where the software has been established as a configurable item.

3.2747
software configuration management (SCM)
1. the 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

3.2748
software design
1. the 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

3.2749
software design audit
1. a review of a software product to determine compliance with requirements, standards, and contractual documents

3.2750
software design concept
1. a fundamental idea (such as information hiding) that can be applied to designing a system

3.2751
software design description (SDD)
1. a representation of software created to facilitate analysis, planning, implementation, and decision-making. *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*.3.1.28; *IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions*.3.4

NOTE The software design description is used as a medium for communicating software design information and may be thought of as a blueprint or model of the system.

3.2752
software design notation
1. a means of describing a software design. *Syn:* software design representation

EXAMPLE structure charts and pseudocode

NOTE It can be diagrammatic, symbolic, or textual.

3.2753
software design verification
1. the evaluation of a design to determine correctness with respect to stated requirements, conformance to design standards, system efficiency, and other criteria

3.2754
software development cycle
1. the 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 This cycle typically includes a requirements phase, design phase, implementation phase, test phase, and sometimes, installation and checkout phase. The phases listed above may 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.

3.2755
software development file (SDF)
1. a collection of material pertinent to the development of a given software unit or set of related units. *Syn:* software development folder, software development notebook, unit development folder

NOTE Contents typically include the requirements, design, technical reports, code listings, test plans, test results, problem reports, schedules, and notes for the units.

3.2756
software development library
1. a software library containing computer readable and human readable information relevant to a software development effort. *Syn:* project library, program support library

cf. master library, production library, software repository, system library

3.2757
software development plan (SDP)
1. a project plan for a software development project

3.2758
software development process
1. the process by which user needs are translated into a software product

NOTE The process involves translating user needs into software requirements, transforming the software requirements into design, implementing the design in code, testing the code, and sometimes, installing and checking out the software for operational use. These activities may overlap or be performed iteratively.

3.2759
software diversity
1. a 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

3.2760
software engineering
1. the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.07. ***2.*** the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software

3.2761
software engineering environment (SEE)
1. environment that provides automated services for the engineering of software systems and related domains, such as project management and process management. *ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services.*2.2.1 ***2.*** hardware, software, and firmware used to perform a software engineering effort

NOTE It includes the platform, system software, utilities, and CASE tools installed.

3.2762
software feature
1. a software characteristic specified or implied by requirements documentation*. IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

EXAMPLE functionality, performance, attributes, or design constraints

3.2763
software hazard
1. a software condition that is a prerequisite to an accident*. IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.5

3.2764
software header
1. information about a software file to facilitate its management, embedded within the file itself*. ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 13

NOTE The software header is one type of information within the more generic category of software tag information.

3.2765
software integrity level
1. the integrity level of a software item*. ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.16

3.2766
software item
1. identifiable part of a software product*. ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software.*3.14. ***2.*** an 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 ***3.*** source code, object code, control code, control data, or a collection of these items*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.41

3.2767
software library
1. a controlled collection of software and related documentation designed to aid in software development, use, or maintenance. ***2.*** a 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. *Syn:* program library

NOTE Types include master library, production library, software development library, software repository, system library.

3.2768
software life cycle (SLC)
1. the project-specific sequence of activities that is created by mapping the activities of this standard onto a selected software life cycle model (SLCM)*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*3.1.13. ***2.*** the system or product cycle initiated by a user need or a perceived customer need and terminated by discontinued use of the product*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2. ***3.*** the period of time that begins when a software product is conceived and ends when the software is no longer available for use

3.2769
software maintenance
1. the totality of activities required to provide cost-effective support to a software system*. ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.1

NOTE Pre-delivery activities include planning for post-delivery operations, supportability, and logistics determination. Post-delivery activities include software modification, training, and operating a help desk.

3.2770
software monitor
1. a software tool that executes concurrently with another program and provides detailed information about the execution of the other program

cf. hardware monitor, monitor

3.2771
software package
1. a complete and documented set of programs supplied to several users for a generic application or function. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.05

NOTE Some software packages are alterable for a specific application.

3.2772
software piracy
1. illegal use or copying of software products. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.07.05

3.2773
software product
1. set of computer programs, procedures, and possibly associated documentation and data. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.42. ***2.*** any of the individual items in (1). ***3.*** One or more computer programs together with any accompanying ancillary nonelectronic, non-mechanical items such as documentation and worksheets, delivered under a single name for use by others. *IEEE Std 1063-2001 (R2007) IEEE Standard for Software User Documentation.*2.1

NOTE Products include intermediate products and products intended for users, such as developers and maintainers.

3.2774
software product developer
1. the 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

3.2775
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 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.50

NOTE This definition can be compared to that of testing in ISO/IEC Guide 2. However, the term evaluation is preferred in order to avoid confusion with the notion of testing widely accepted in the field of software engineering. Software product evaluation is not necessarily conformity testing (as defined in ISO/IEC Guide 2) in the context of a certification scheme. However, conformity testing can be part of an evaluation.

3.2776
software project
1. the set of work activities, both technical and managerial, required to satisfy the terms and conditions of a project agreement. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.6

NOTE A software project should have specific starting and ending dates, well-defined objectives and constraints, established responsibilities, and a budget and schedule. A software project may be self-contained or may be part of a larger project. In some cases, a software project may span only a portion of the software development cycle. In other cases, a software project may span many years and consist of numerous subprojects, each being a well-defined and self-contained software project.

3.2777
software project life cycle (SPLC)
1. the portion of the entire software life cycle applicable to a specific project. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.1.1

NOTE It is the sequence of activities created by mapping the activities of IEEE Std 1074 onto a selected software project life cycle model (SPLCM).

3.2778
software project life cycle model (SPLCM)
1. the framework selected by each using organization on which to map the activities of IEEE Std 1074 to produce the software project life cycle (SPLC). *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.1.2. *Syn:* software project life cycle

3.2779
software project life cycle process (SPLCP)
1. the project-specific description of the process developed by adding the organizational process assets (OPAs) to the software project life cycle (SPLC) and the OPAs. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process. Syn:* software project life cycle

3.2780
software quality
1. capability of a software product to satisfy stated and implied needs when used under specified conditions. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.51

NOTE This definition differs from the ISO 9000:2000 quality definition mainly because the software quality definition refers to the satisfaction of stated and implied needs, while the ISO 9000 quality definition refers to the satisfaction of requirements. In SQuaRE standards software quality has the same meaning as software product quality.

3.2781
software quality characteristic
1. category of software quality attributes that bears on software quality. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.52

NOTE Software quality characteristics may be refined into multiple levels of sub-characteristics and finally into software quality attributes.

3.2782
software quality evaluation
1. systematic examination of the extent to which a software product is capable of satisfying stated and implied needs. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.53

3.2783
software quality in use
1. capability of a software product to enable specific users to achieve specific goals with effectiveness, productivity, safety and satisfaction in specific contexts of use. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.54

NOTE Before the product is released, quality in use can be specified and measured in a test environment for the intended users, goals and contexts of use. Once in use, it can be measured for actual users, goals and contexts of use. The actual needs of users may not be the same as those anticipated in requirements, so actual quality in use may be different from quality in use measured earlier in a test environment.

3.2784
software quality measure
1. measure of internal software quality, external software quality or software quality in use. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.55

NOTE Internal software quality, external software quality and software quality in use are described in the quality model in ISO/IEC 9126-1 [ISO/IEC 25010].

3.2785
software quality metric
1. a 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 Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.24

3.2786
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

cf. software configuration management, version

3.2787
software reliability
1. the probability that software will not cause the failure of a system for a specified time under specified conditions

NOTE The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered.

3.2788
software reliability management
1. the 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

3.2789
software repository
1. a software library providing permanent, archival storage for software and related documentation

cf. master library, production library, software development library, system library

3.2790
software requirement
1. a software capability needed by a user to solve a problem to achieve an objective. ***2.*** a 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

3.2791
software requirements analysis
1. the process of studying user needs to arrive at a definition of system, hardware, or software requirements

3.2792
software requirements engineering
1. the science and discipline concerned with analyzing and documenting software requirements. ***2.*** software requirements elicitation, analysis, specification, verification, and management

NOTE It involves transforming system requirements into a description of software requirements, performance parameters, and a software configuration using an iterative process of definition, analysis, trade-off studies, and prototyping.

3.2793
software requirements management
1. the 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

3.2794
software requirements phase
1. the software development life-cycle phase during which the requirements for a software product, such as functional and performance capabilities, are defined, documented, and reviewed

3.2795
software requirements review (SRR)
1. a 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.*** a review as in (1) for any software component

cf. system requirements review

NOTE This review is called software specification review by the US Department of Defense.

3.2796
software requirements specification (SRS)
1. documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.29

3.2797
software requirements verification
1. the 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

3.2798
software safety
1. freedom from software hazards*. IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans*.3.1.6

3.2799
software safety program
1. a systematic approach to reducing software risks*. IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans*.3.1.7

3.2800
software specification review (SSR)
1. a joint acquirer-supplier review conducted to finalize software configuration item (SCI) requirements so that the software developer can initiate the next step in the software development process

NOTE The SSR is conducted when SCI requirements have been sufficiently defined to evaluate the developer's responsiveness to and interpretation of the system- or segment-level technical requirements. A successful SSR is predicated on the developer's determination that the software requirements specification and interface specifications form a satisfactory basis for proceeding to the preliminary design phase.

3.2801
software support
1. the act of maintaining the software and its associated documentation in a functional state*. ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.7

NOTE Software support may be given by the manufacturer, marketing organization, supplier or other organization. In special contractually-agreed cases, consumers may be permitted to maintain or enhance the software themselves.

3.2802
software system
1. a software-intensive system for which software is the only component to be developed or modified*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2

3.2803
software tag
1. information about a software file or package to facilitate its management, some of which information may be held within a software header*. ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 14

3.2804
software test environment (STE)
1. the facilities, hardware, software, firmware, procedures, and documentation needed to perform qualification or other testing of software

NOTE Elements may include but are not limited to simulators, code analyzers, test case generators, and path analyzers, and may also include elements used in the software engineering environment.

3.2805
software test incident
1. an event occurring during the execution of a software test that requires investigation*. IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

3.2806
software testing
1. the dynamic verification of the behavior of a program on a finite set of test cases, suitably selected from the usually infinite executions domain, against the expected behavior*. ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*5

3.2807
software tool
1. a computer program used in the development, testing, analysis, or maintenance of a program or its documentation. ***2.*** a 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, flowcharter, monitor, test case generator, timing analyzer

3.2808
software transition
1. a controlled and coordinated sequence of actions wherein software development passes from the organization performing initial software development to the organization performing software maintenance*. ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance.*3.1

3.2809
software unit
1. separately compilable piece of code*. ISO/IEC 12207:2008 (IEEE Std 12207-2008); Systems and software engineering — Software life cycle processes.*4.43. ***2.*** the lowest element in one or more software components*. IEEE/EIA 12207.2-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:199,; Standard for Information Technology — Software Life Cycle Processes — Implementation considerations.*3.2

3.2810
software user documentation
1. Electronic or printed body of material that provides information to users of software*. IEEE Std 1063-2001 (R2007) IEEE Standard for Software User Documentation.*2.11

3.2811
software-based system
1. computer system controlled by software*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.33

3.2812
software-intensive system
1. a system for which software is a major technical challenge and is perhaps the major factor that affects system schedule, cost, and risk*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.18

NOTE In the most general case, a software-intensive system is comprised of hardware, software, people, and manual procedures.

3.2813
solution domain
1. the environment in which a solution or set of solutions resides*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.21

3.2814
sort
1. a 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

3.2815
source
1. an item or activity having a potential for a consequence*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.18

NOTE In the context of safety, source is a hazard (refer to ISO/IEC Guide 51:1999).

3.2816
source address
1. the address of a device or storage location from which data is to be transferred

cf. destination address

3.2817
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 A source program is made up of source code.

3.2818
source language
1. the 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

3.2819
source program
1. a computer program that must be compiled, assembled, or otherwise translated in order to be executed by a computer

cf. object program

3.2820
source statements (SS)
1. the encoded logic of the software product

NOTE Source statements may be classified by function as executable, data declaration, compiler directive, or comment. They may also be classified as deliverable or nondeliverable.

3.2821
SOW
1. statement of work. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2822
special cause
1. a source of variation that is not inherent in the system, is not predictable, and is intermittent. It can be assigned to a defect in the system. On a control chart, points beyond the control limits, or non-random patterns within the control limits, indicate it. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* assignable cause

cf. common cause

3.2823
specialize
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.181

cf. respecialize; unspecialize

NOTE A specialized instance acquires a different (lower) lowclass.

3.2824
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

3.2825
specification
1. a detailed formulation, in document form, which provides a definitive description of a system for the purpose of developing or validating the system. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.03. ***2.*** a document that fully describes a design element or its interfaces in terms of requirements (functional, performance, constraints, and design characteristics) and the qualification conditions and procedures for each requirement. *IEEE Std 1220-2005 IEEE Standard for the*

*Application and Management of the Systems Engineering Process.*3.1.28. ***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) — Fourth Edition*

3.2826
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

3.2827
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

3.2828
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) — Fourth Edition*

cf. control limits

3.2829
specification tree
1. a diagram that depicts all of the specifications for a given system and shows their relationships to one another. ***2.*** a hierarchy of specification elements and their interface specifications that identify the elements and the specifications related to design elements of the system configuration that are to be controlled*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.29

3.2830
SPI
1. schedule performance index*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2831
spiral model
1. a 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

3.2832
SPLC
1. software project life cycle*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2833
SPLCM
1. software project life cycle model*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2834
SPLCP
1. software project life cycle process*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2835
split key
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.183

NOTE [key style]

3.2836
SPM
1. Software Project Management*. ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5

3.2837
SPMPI
1. Software Project Management Planned Information*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2838
sponsor
1. the person or group that provides the financial resources, in cash or in kind, for the project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2839
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 derived from the acronym SPOOL for Simultaneous Peripheral Output On Line

3.2840
spooler
1. a program that initiates and controls spooling

3.2841
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

3.2842
spreadsheet program
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.21

3.2843
SQA
1. Software Quality Assurance (Group)*. ISO/IEC TR 9126-3:2003, Software engineering — Product quality — Part 3: Internal metrics.*5

3.2844
SQL
1. Structured Query Language

3.2845
squiggle
1. a short ""S""-shaped line attached at one end to an arrow label and at the other end to an arrow segment. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.101

NOTE A squiggle binds an object type set (arrow label) to an object set (arrow segment).

3.2846
SRMPI
1. Software Release Management Planned Information. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.2

3.2847
SRR
1. software requirements review. ***2.*** system requirements review

3.2848
SRS
1. software requirements specification. ***2.*** system requirements specification. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.2

3.2849
SS
1. start-to-start. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2850
SSR
1. software specification review

cf. software requirements review

3.2851
stability schema
1. a specification of failure modes which an object will not exhibit. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*16.2.1.1

3.2852
stabilization phase
1. the time interval of the measurement procedure when the RTE starts submitting tasks until the SUT reaches a stable state of operation. *ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.16

3.2853
stable branch
1. a branch where stability-disrupting changes are discouraged

NOTE the branch used for releasing the product's stable production version

3.2854
stable process
1. a 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

3.2855
staff-hour
1. an hour of effort expended by a member of the staff

3.2856
staffing management plan
1. the document that describes when and how human resource requirements will be met. It is contained in, or is a subsidiary plan of, the human resource plan*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2857
stage
1. period within the life cycle of an entity that relates to the state of its description or realization*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.44; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.28

NOTE Stages relate to major progress and achievement milestones of the system through its life cycle. Stages may be overlapping.

3.2858
staged representation
1. a structure wherein attaining the goals of a set of process areas establishes a maturity level; each level builds a foundation for subsequent levels

3.2859
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 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.45; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.29; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.37. ***2.*** individual, group or organization that can affect, be affected by, or perceive itself to be affected by, a risk*. ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management.*3.19. ***3.*** individual, group, or organization who may affect, be affected by, or perceive itself to be affected by a decision or activity*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.16 ***4.*** person or organization (e.g., customer, sponsor, performing organization, or the public) that is actively involved in the project, or whose interests may be positively or negatively affected by execution or completion of the project. A stakeholder may also exert influence over the project and its deliverables*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

EXAMPLE end users, end user organizations, supporters, developers, producers, trainers, maintainers, disposers, acquirers, supplier organizations and regulatory bodies

NOTE The decision-maker is also a stakeholder.

3.2860
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

3.2861
standard
1. set of mandatory requirements established by consensus and maintained by a recognized body to prescribe a disciplined uniform approach or specify a product, that is, mandatory conventions and practices ***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) — Fourth Edition*

3.2862
standard process
1. the set of definitions of the basic processes that guide all processes in an organization. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary*.3.50

NOTE These process definitions cover the fundamental process elements (and their relationships to each other) that must be incorporated into the defined processes that are implemented in projects across the organization. A standard process establishes consistent activities across the organization and is desirable for long-term stability and improvement. The organization's set of standard processes describes the fundamental process elements that will be part of the projects' defined processes. It also describes the relationships (for example, ordering and interfaces) between these process elements.

3.2863
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

3.2864
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) — Fourth Edition*

3.2865
starting address
1. the address of the first instruction of a computer program in main storage

cf. origin, assembled origin, loaded origin

NOTE This address may or may not be the same as the program's origin, depending upon whether there are data preceding the first instruction.

3.2866
start-to-finish (SF)
1. the logical relationship where completion of the successor schedule activity is dependent upon the initiation of the predecessor schedule activity. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. logical relationship

3.2867
start-to-start (SS)
1. the logical relationship where initiation of the work of the successor schedule activity depends upon the initiation of the work of the predecessor schedule activity. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

cf. logical relationship

3.2868
state
1. a condition or mode of existence that a system, component, or simulation may be in. ***2.*** the values assumed at a given instant by the variables that define the characteristics of a system, component, or simulation. ***3.*** the 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.*** a condition that characterizes the behavior of a function/subfunction or element at a point in time. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process*.3.1.32

3.2869
state class
1. a class that represents a set of real or abstract objects that have common knowledge or behavior. *IEEE Std 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 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, i.e., their property values can change.

3.2870
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. *IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

3.2871
state diagram
1. a 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

3.2872
state name
1. the 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

3.2873
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

3.2874
statement of work (SOW)
1. a narrative description of products, services, or results to be supplied. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition.* ***2.*** document used by the acquirer to describe and specify the tasks to be performed under the contract. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.46

3.2875
statement testing
1. testing designed to execute each statement of a computer program

cf. branch testing, path testing

3.2876
static
1. pertaining to an event or process that occurs without computer program execution

cf. dynamic

EXAMPLE static analysis, static binding

3.2877
static analysis
1. the process of evaluating a system or component based on its form, structure, content, or documentation

cf. dynamic analysis, inspection, walk-through

3.2878
static binding
1. binding performed prior to the execution of a computer program and not subject to change during program execution

cf. dynamic binding

3.2879
static breakpoint
1. a 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

3.2880
static dump
1. a 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

3.2881
static error
1. an error that is independent of the time-varying nature of an input

cf. dynamic error

3.2882
static model
1. a model that describes an interrelated set of classes (and/or subject domains) along with their relationships and responsibilities. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.185

cf. dynamic model

3.2883
static schema
1. a 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 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*6.1.2

NOTE Thus, a static schema is the specification of the types of one or more information objects at some particular point in time. These types are subtypes of the types specified in the invariant schema.

3.2884
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

3.2885
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

3.2886
status code
1. a code used to indicate the results of a computer program operation. *Syn:* condition code

EXAMPLE a code indicating a carry, an overflow, or a parity error

3.2887
step
1. one element (numbered list item) in a procedure that tells a user to perform an action (or actions). *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.47. ***2.*** the 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.*** an abstraction of an action, used in a process, that may leave unspecified objects that participate in that action. *ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.3.6

NOTE A step contains one or more actions. Responses by the software are not considered to be steps.

3.2888
stepwise refinement
1. a 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

3.2889
stop
1. to terminate the execution of a computer program. *Syn:* halt (1)

cf. pause

3.2890
storage
1. a functional unit into which data can be placed, in which data can be retained, and from which data can be retrieved. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.10. *Syn:* storage device

3.2891
storage allocation
1. an 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

3.2892
storage capacity
1. the maximum number of items that can be held in a given storage device; usually measured in words or bytes

cf. channel capacity, memory capacity

3.2893
storage efficiency
1. the degree to which a system or component performs its designated functions with minimum consumption of available storage

cf. execution efficiency

3.2894
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

3.2895
straight-line code
1. a sequence of computer instructions in which there are no loops

3.2896
straight-line coding
1. a 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

3.2897
strategy
1. an organization's overall plan of development, describing the effective use of resources in support of the organization in its future activities*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.17

NOTE involves setting objectives and proposing initiatives for action

3.2898
stratified language
1. a language that cannot be used as its own metalanguage

cf. unstratified language

EXAMPLE FORTRAN, COBOL

3.2899
stream interface
1. an interface in which all the interactions are flows*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.8

3.2900
stream interface signature
1. an interface signature for a stream interface*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*7.1.13

NOTE 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.

3.2901
strengths, weaknesses, opportunities, and threats (SWOT) analysis
1. this information gathering technique examines the project from the perspective of each project's strengths, weaknesses, opportunities, and threats to increase the breadth of the risks considered by risk management*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2902
stress testing
1. testing conducted to evaluate a system or component at or beyond the limits of its specified requirements

cf. boundary value

3.2903
strong typing
1. a 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

3.2904
structural testing
1. testing that takes into account the internal mechanism of a system or component. *Syn:* glass-box testing, white-box testing

cf. functional testing (1)

NOTE Types include branch testing, path testing, statement testing.

3.2905
structure chart
1. a 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. *Syn:* hierarchy chart, program structure chart

cf. call graph

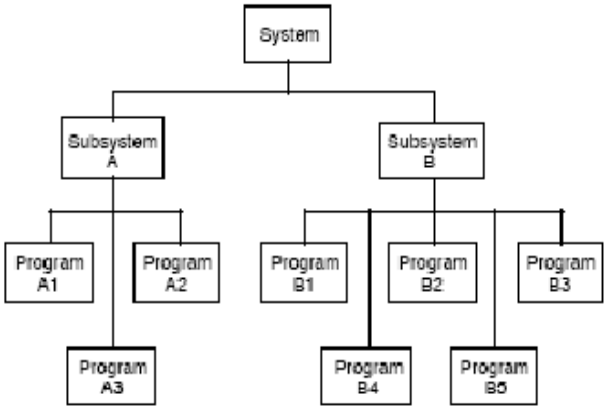


Figure 17 — Structure chart

NOTE The result is not necessarily the same as that shown in a call graph.

3.2906
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

3.2907
structured design
1. a 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.*** the result of applying the approach in (1)

cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping

3.2908
structured program
1. a computer program constructed of a basic set of control structures, each having one entry and one exit

cf. structured design

NOTE The set of control structures typically includes: sequence of two or more instructions, conditional selection of one of two or more sequences of instructions, and repetition of a sequence of instructions.

3.2909
structured programming
1. a software development technique that includes structured design and results in the development of structured programs

3.2910
structured programming language
1. a 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

3.2911
structured walkthrough
1. a systematic examination of the requirements, design, or implementation of a system, or any part of it, by qualified personnel. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.03.04

3.2912
stub
1. a 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.*** a computer program statement substituting for the body of a software module that is or will be defined elsewhere. ***3.*** an 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 10746-3:1996, 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

3.2913
style
1. set of language-specific editorial conventions covering grammar, terminology, punctuation, capitalization, and word choice of documentation. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.48

3.2914
subclass
1. a specialization of one or more superclasses. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.186. *Syn:* subtype

NOTE 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.

3.2915
subclass cluster
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.187. ***2.*** a set of one or more mutually exclusive specializations of the same generic entity. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.187. *Syn:* category cluster

NOTE A cluster exists when an instance of the superclass can be an instance of only one of the subclasses in the set, and each instance of a subclass is an instance of the superclass.

3.2916
subclass responsibility
1. a designation that a property of a class must be overridden in its subclasses. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.188

NOTE That is, the designation given to a property whose implementation is not specified in this class. A property that is a subclass responsibility is a specification in the superclass of an interface that each of its subclasses must provide. A property that is designated as a subclass responsibility has its realization deferred to the subclass(es) of the class.

3.2917
subject area
1. a related collection of meta-object instance definitions. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE Subject areas are used to define scoped areas of interest. Subject areas overlap to ensure the integration of the overall metamodel, but a tool need only use those subject areas relevant to the data to be exported or imported.

3.2918
subject domain
1. an area of interest or expertise. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.189

NOTE The responsibilities of a subject domain are an aggregation of the responsibilities of a set of current or potential named classes. A subject domain may also contain other subject domains. A subject domain encapsulates the detail of a view.

3.2919
subject domain responsibility
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.190

NOTE The classes and subject domains in a subject domain together supply the knowledge, behavior, and rules that make up the subject. These notions are collectively referred to as the subject domain's responsibilities. Subject domain responsibilities are not distinguished as sub-domains or classes during the early stages of analysis.

3.2920
subject system
1. a computing system (existing or to be created) about which descriptive information is being developed in a computing system or CASE tool. *IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnection — Classification and Description.*3.1

3.2921
subject system
1. a computing system (existing or to be created) about which descriptive information is being developed in a computing system or CASE tool. *IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.13

3.2922
subject tool
1. a particular computing system tool or CASE tool that is the focus for a description of interconnections and tool content. *IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnections-Classification and Description.*3.11; *IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.14

3.2923
submit primitive
1. a 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

3.2924
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) — Fourth Edition*

3.2925
subordinate web site
1. a site contained within another web site. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.10

3.2926
subphase
1. a subdivision of a phase. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2927
subprogram
1. a separately compilable, executable component of a computer program

cf. coroutine, main program, routine, subroutine

NOTE The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages.

3.2928
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) — Fourth Edition*

3.2929
subroutine
1. a routine that returns control to the program or subprogram that called it

cf. coroutine, closed subroutine, open subroutine

NOTE The terms 'routine,' 'subprogram,' and 'subroutine' are defined and used differently in different programming languages.

3.2930
subroutine trace
1. a 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. *Syn:* call trace

cf. execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace

3.2931
substitutability
1. a principle stating that, since each instance of a subclass is an instance of the superclass, an instance of the subclass should be acceptable in any context where an instance of the superclass is acceptable. *IEEE Std*

1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).3.1.191

NOTE Any request sent to an instance receives an acceptable response, regardless of whether the receiver is an instance of the subclass or the superclass.

3.2932
subsystem
1. a secondary or subordinate system within a larger system. ***2.*** a system that is part of a larger system. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.17

3.2933
subtype
1. a subset of a data type, obtained by constraining the set of possible values of the data type. ***2.*** a 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

cf. derived type

NOTE The operations applicable to the subtype are the same as those of the original data type.

3.2934
successful adoption
1. the 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

3.2935
successor activity
1. the schedule activity that follows a predecessor activity, as determined by their logical relationship. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2936
summary activity
1. a group of related schedule activities aggregated at some summary level, and displayed/reported as a single activity at that summary level. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition. Syn:* hammock activity

cf. subproject, subnetwork

3.2937
superclass
1. a class whose instances are specialized into one or more subclasses. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.193. *Syn:* supertype

cf. partial cluster, total cluster

3.2938
supercomputer
1. the class of computers that have the highest processing speeds available at a given time. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.24

NOTE often used for solving scientific and engineering problems

3.2939
supervisor state
1. in the operation of a computer system, a state in which the supervisory program is executing. *Syn:* executive state, master state, privileged state

cf. problem state

NOTE This state usually has higher priority than, and precludes the execution of, application programs.

3.2940
supervisory program
1. a 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. *Syn:* control program, executive, executive program, supervisor

cf. supervisor state

3.2941
supplementary run
1. the 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

3.2942
supplier
1. organization or individual that enters into an agreement with the acquirer for the supply of a product or service. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.47; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.3; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.38. ***2.*** an organization that develops some or all of the project deliverables for an acquirer. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.7. ***3.*** the organization that sells the software package to the consumer. *ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages.*3.2.6. ***4.*** a person or organization that enters into a contract with the acquirer for the supply of a software product (which may be part of a system) under the terms of the contract. *IEEE Std 1062, 1998 Edition (R2002) IEEE Recommended Practice for Software Acquisition (includes IEEE Std 1062a).*3.12. ***5.*** the person, or persons, who produce a product for a customer. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.*3.3. *Syn:* contractor, producer, seller, vendor

NOTE Suppliers may include organizations that have primary responsibility for project deliverables and subcontractors that deliver some part of the project deliverables to a primary supplier. In the latter case, the primary supplier is also an acquirer. The acquirer may designate a part of its organization as supplier.

3.2943
support
1. the 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

3.2944
support activity group
1. an 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. *IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.1.4

3.2945
support manual
1. a document that provides the information necessary to service and maintain an operational system or component throughout its life cycle. *Syn:* maintenance manual

cf. diagnostic manual, installation manual, operator manual, programmer manual, user manual

NOTE Typically described are the hardware and software that make up the system or component and procedures for servicing, repairing, or reprogramming it.

3.2946
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.03

cf. application software, system software

EXAMPLE compilers, loaders, and other utilities

3.2947
support staff-hour
1. an 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

3.2948
supporting data item
1. data used to describe an anomaly and the environment in which it was encountered. *IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies.*3.7

3.2949
supporting process
1. a collection of work activities that span the entire duration of a software project. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.8

EXAMPLE software documentation, quality assurance, configuration management, software reviews, audit processes, and problem resolution activities

3.2950
sustainment
1. activities performed to ensure that a product or service remains operational

cf. maintenance

3.2951
SV
1. schedule variance. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2952
swap
1. an 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

3.2953
SWOT
1. strengths, weaknesses, opportunities, and threats. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.2954
symbol
1. a graphic representation of a concept that has meaning in a specific context. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.02.07

3.2955
symbol table
1. a table that presents program symbols and their corresponding addresses, values, and other attributes

3.2956
symbolic address
1. an 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

3.2957
symbolic execution
1. a 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

3.2958
symbolic language
1. a 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

3.2959
symbolic trace
1. a 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

3.2960
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

3.2961
synchronous
1. pertaining to two or more processes that depend upon the occurrence of specific events such as common timing. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.28

3.2962
synchronous message communication
1. a form of communication in which a producer task sends a message to a consumer task and waits for acknowledgment. *Syn:* tightly coupled message communication

3.2963
synchronous message communication with reply
1. a form of communication in which a producer (or client) task sends a message to a consumer (or server) task and waits for a reply. *Syn:* tightly coupled message communication with reply

3.2964
synchronous message communication without reply
1. a form of communication in which a producer task sends a message to a consumer task and waits for the consumer to accept the message. *Syn:* tightly coupled message communication without reply

3.2965
synchronous request
1. a request where the client pauses to wait for completion of the request. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.17

3.2966
syntactic agreement
1. a passive interconnection in which two things agree on a set of symbols and symbol arrangements (statements) by which they will communicate. *IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.15

cf. semantic agreement

3.2967
syntactic error
1. a violation of the structural or grammatical rules defined for a language. *Syn:* syntax error

cf. semantic error

EXAMPLE using the statement B + C = A in FORTRAN, rather than the correct A = B + C

3.2968
syntax
1. the 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.*** the structural components or features of a language and rules that define the ways in which the language constructs may be assembled together to form sentences. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.195. ***3.*** a 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

3.2969
SYNTAX.1
1. the primary syntax defined within the CDIF family of standards. *ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE The CDIF family of standards supports multiple transfer formats, each composed of a syntax and an encoding.

3.2970
system
1. combination of interacting elements organized to achieve one or more stated purposes. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.31; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.39;*ISO/IEC TR 90005:2008, Systems engineering — Guidelines for the application of ISO 9001 to system life cycle processes.*2.1. ***2.*** an interdependent group of people, objects, and procedures constituted to achieve defined objectives or some operational role by performing specified functions. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.16. ***3.*** a collection of interacting components organized to accomplish a specific function or set of functions within a specific environment. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.2. ***4.*** an interacting combination of elements to accomplish a defined objective. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*2.1.6. ***5.*** a set or arrangement of elements that are related, and whose behavior satisfies operational needs and provides for the life cycle sustainment of the products. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.34. ***6.*** a conceptual entity defined by its boundaries. ***7.*** 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 A system may be 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' may be substituted simply by a context-dependent synonym, e.g., aircraft, though this may then obscure a system principles 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.

3.2971
system analysis
1. a 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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.05. *Syn:* systems analysis

3.2972
system architecture
1. the composite of the design architectures for products and their life cycle processes. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.35

3.2973
system breakdown structure (SBS)
1. a hierarchy of elements, related life cycle processes, and personnel used to assign development teams, conduct technical reviews, and to partition out the assigned work and associated resource allocations to each of the tasks necessary to accomplish the objectives of the project. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.39

NOTE It also provides the basis for cost tracking and control.

3.2974
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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.04

3.2975
system design
1. a process of defining the hardware and Software architecture, components, modules, interfaces and data for a system to satisfy specified requirements. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.03.01

3.2976
system design review (SDR)
1. a review conducted to evaluate the manner in which the requirements for a system have been allocated to configuration items, the system engineering process that produced the allocation, the engineering planning for the next phase of the effort, manufacturing considerations, and the planning for production engineering

cf. critical design review, preliminary design review

3.2977
system development
1. a process that usually includes requirements analysis, system design, implementation, documentation and quality assurance. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.01

3.2978
system development cycle
1. the 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 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.

3.2979
system documentation
1. the collection of documents that describe the requirements, capabilities, limitations, design, operation, and maintenance of an information processing system. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.04

3.2980
system effectiveness
1. a measurement of the ability of a system to satisfy its intended operational uses as a function of how the system performs under anticipated environmental conditions, and the ability to produce, test, distribute, operate, support, train, and dispose of the system throughout its life cycle. *IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.40

3.2981
system element
1. member of a set of elements that constitutes a system. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.49; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.32

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 A system element is a discrete part of a system that can be implemented to fulfill specified requirements.

3.2982
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

3.2983
system follow-up
1. the study of the effects of a system after it has reached a stabilized state of operational use. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.02.09. *Syn:* post-implementation review, post-development review

3.2984
system hazard
1. a system condition that is a prerequisite to an accident. *IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.8

3.2985
system integration
1. the progressive assembling of system components into the whole system. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.04.02

3.2986
system interface task
1. a task that hides the interface to and communicates with an external system or subsystem

3.2987
system library
1. a 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

3.2988
system life cycle
1. the course of developmental changes through which a system passes from its conception to the termination of its use. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.05. ***2.*** the period that begins when a system is conceived and ends when the system is no longer available for use

3.2989
system maintenance
1. the modification of a system to correct faults, to improve performance, or to adapt the system to a changed environment or changed requirements. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.09

3.2990
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

3.2991
system of systems
1. a large system that delivers unique capabilities, formed by integrating independently useful systems

3.2992
system profile
1. a 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

3.2993
system requirements review (SRR)
1. a 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

3.2994
system requirements specification (SyRS)
1. a structured collection of information that embodies the requirements of the system. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.17

cf. software requirements specification, SRS

3.2995
system safety
1. freedom from system hazards. *IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans.*3.1.9

3.2996
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.04.02

cf. application software, support software

EXAMPLE operating systems, assemblers, utilities

3.2997
system stakeholder
1. an individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system. *IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*3.8

3.2998
system support
1. the continued provision of services and material necessary for the use and improvement of an implemented system. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.12

3.2999
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*

3.3000
system testing
1. testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.37

3.3001
system under test (SUT)
1. the parts of the CBSS to be tested. *ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.18

NOTE The SUT may consist of hardware, system software, data communication features or application software or a combination of them.

3.3002
systematic failure
1. a failure related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors. *ISO/IEC 15026:1998, Information technology — System and software integrity levels.*3.19

3.3003
systematic reuse
1. the practice of reuse according to a well-defined, repeatable process. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.*3.19

3.3004
system-of-interest
1. the system whose life cycle is under consideration in the context of this International Standard. *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.33. *Syn:* system of interest

3.3005
systems engineering
1. interdisciplinary approach governing the total technical and managerial effort required to transform a set of customer needs, expectations, and constraints into a solution and to support that solution throughout its life

cf. hardware engineering, software engineering

NOTE 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

3.3006
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. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.1.36

3.3007
T&M
1. time and material. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3008
table
1. a more concrete representation of an entity. *ISO/IEC 15476-4:2005, Information technology — CDIF semantic metamodel — Part 4: Data models.*6.3

3.3009
table heading
1. the symbolic name or other means of referencing a decision table from other documents. *ISO 5806:1984, Information processing — Specification of single-hit decision tables.*3.12

NOTE alternatively, or in addition, a clear description of the table

3.3010
table of contents
1. list of the headings in a document in page number order, with page numbers shown against each heading. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.45

3.3011
table of effective pages
1. list showing the latest version number of each page in a loose-leaf paper document. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.46

NOTE Where individual pages are replaced, the table of effective pages shows the old version number for the unaltered pages, and the new version number for the replaced pages.

3.3012
table-driven method
1. a scheme that lets a program look up information in a table rather than using logic statements

3.3013
tag
1. a symbolic name assigned to a specific release or a branch

NOTE provides developers and end users with a unique reference to the code base they are working with

3.3014
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

3.3015
tailoring
1. omitting tasks outside the scope of work of the relevant contract. *IEEE/EIA 12207.2-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Implementation considerations.*1.2

3.3016
tailoring guideline
1. instructions that enable an organization to adapt the process description of standard processes appropriately to meet specific needs. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.52

NOTE Tailoring a process adapts the process description for a particular end. For example, a project creates its defined process by tailoring the organization's set of standard processes to meet the objectives, constraints, and environment of the project. The organization's set of standard processes is described at a general level that may not be 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.

3.3017
target capability
1. the process capability which the process capability determination sponsor judges will represent an acceptable process risk to the successful implementation of the specified requirement. *ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.54

3.3018
target language
1. the language in which the output from a machine-aided translation process is represented. *Syn:* object language

cf. source language

EXAMPLE the language output by an assembler or compiler

3.3019
target machine
1. the computer on which a program is intended to execute. ***2.*** a computer being emulated by another computer

cf. host machine (1)

3.3020
target of process
1. software product or task executed by software product to which measurement or evaluation process is applied. *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.59

3.3021
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 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.34; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.34. ***2.*** in software design, a software component that can operate in parallel with other software components. ***3.*** the activities required to achieve a goal. *ISO/IEC TR 9126-4:2004, Software engineering — Product quality — Part 4: Quality in use metrics.*4.3. ***4.*** a concurrent object with its own thread of control. ***5.*** a 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. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.38

NOTE Related tasks are usually grouped to form activities.

3.3022
task behavior specification
1. a specification describing a concurrent task's interface, structure, timing characteristics, relative priority, errors detected, and task event sequencing logic

3.3023
task completion
1. timely event when for a specific task the total output string or, in case of a set of output strings, all parts are completely received by to the emulated user or another instance*. ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.20

NOTE The time of task completion defines the end time of the preceding preparation time and the begin time of the execution time of the following task.

3.3024
task interface
1. input or output, events signaled (input or output), external inputs or outputs, or access to passive objects

3.3025
task inversion
1. an optimization concept whereby the tasks in a system can be merged in a systematic way

3.3026
task mode
1. indication of whether the user's preparation time begins immediately with the task submission of the preceding task (value = 0, i.e., "NO WAIT") or begins when the preceding task has been completed (task completion) (value = 1, i.e., "WAIT")*. ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.21

NOTE mode of ""Dialog"" or ""Batch"" in UNIX-based systems

3.3027
task priority criteria
1. a category of the task-structuring criteria addressing the relative importance of executing a given task

3.3028
task structuring
1. a software design stage with the objective of structuring a concurrent application into concurrent tasks and defining the task interfaces

3.3029
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 may start, regardless if the SUT starts the execution immediately or not*. ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.22

NOTE Normally the task submission is defined internally by the submission of a special character (e.g. Carriage Return) or a character sequence at the end of the input string or at the end of several parts of the input string. Also it often happens that the task submission event is defined by the submission of the last character of any specified number characters in a string. For a classic batch task the task submission may be defined by the submission of the last character of the last string of the batch command sequence.

3.3030
task type
1. a 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 Emulated users submit only these types of tasks to the SUT.

3.3031
task-clustering criteria
1. category of the criteria addressing whether and how to group objects into concurrent tasks

3.3032
task-structuring criteria
1. a set of heuristics for helping a designer structure a system into concurrent tasks

3.3033
taxonomy
1. a scheme that partitions a body of knowledge and defines the relationships among the pieces

NOTE It is used for classifying and understanding the body of knowledge.

3.3034
TCP
1. Transmission Control Protocol. *ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions.*4

3.3035
TCPI
1. to-complete-performance index

3.3036
TCS
1. Transmission Code Set. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*.3.3

3.3037
TCS-C
1. Char Transmission Code Set. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.3038
TCS-W
1. Wchar Transmission Code Set. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.3039
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

3.3040
team selection plan
1. document specifying the qualifications, experience and training needs of documentation development staff. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.47

3.3041
technical complexity adjustment
1. a factor which attempts to take into account the influence on application size of technical and quality requirements, which may be used to derive the adjusted size. *ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10

NOTE Note that if this is done, the result is not the functional size.

3.3042
technical complexity adjustment factors
1. the set of 19 factors that are taken into account in the technical complexity adjustment (TCA)*. ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual.*10

NOTE Each factor has a degree of influence (DI) between 1 and 5.

3.3043
technical contact
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.49. *Syn:* subject-matter expert, subject matter expert, SME

3.3044
technical management
1. the application of technical and administrative resources to plan, organize, and control engineering functions

3.3045
technical performance measurement
1. [Technique] a performance measurement technique that compares technical accomplishments during project execution to the project management plan's schedule of planned technical achievements. It may use key technical parameters of the product produced by the project as a quality metric. The achieved metric values are part of the work performance information*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3046
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

3.3047
technical review
1. a 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 Std 1028-2008 IEEE Standard for Software Reviews and Audits.*3.7

NOTE Technical reviews may also provide recommendations of alternatives and examination of various alternatives.

3.3048
technical standard
1. a standard that describes the characteristics of applying accumulated technical or management skills and methods in the creation of a product or performing a service

3.3049
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) — Fourth Edition.* ***2.*** methods and skills required to carry out a specific activity*. ISO/IEC 25001:2007, Software engineering — 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

3.3050
technology viewpoint
1. a viewpoint on an ODP system and its environment that focuses on the choice of technology in that system. *ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture*.4.1.1.5

3.3051
template
1. an asset with parameters or slots that can be used to construct an instantiated asset. *IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes*.3.20. ***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) — Fourth Edition*

cf. construction

3.3052
temporal clustering
1. a task-structuring criterion by which activities that are not sequentially dependent, but are activated by the same event are grouped into a task

3.3053
temporal cohesion
1. a 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

3.3054
term
1. an 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

3.3055
term evaluation
1. the 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

3.3056
terminal
1. a functional unit in a system or communication network at which data may be entered or retrieved. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.11

3.3057
terminal symbol
1. a 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

3.3058
termination construct
1. a program construct that results in a halt or exit

3.3059
termination deliver
1. a 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

3.3060
termination submit
1. a 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

3.3061
test
1. an 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.*** to conduct an activity as in (1). ***3.*** a set of one or more test cases and procedures*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.39

3.3062
test approach
1. particular method that will be employed to pick the particular test case values*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.40

NOTE may vary in specificity from very general (e.g., black box or white box) to very specific (e.g., minimum and maximum boundary values)

3.3063
test bed
1. an environment containing the hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test

3.3064
test case
1. a set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.31. ***2.*** documentation specifying inputs, predicted results, and a set of execution conditions for a test item*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.31

3.3065
test case generator
1. a 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. *Syn:* test data generator, test generator

3.3066
test case specification
1. a document specifying inputs, predicted results, and a set of execution conditions for a test item

3.3067
test class
1. designated grouping of test cases*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.42

3.3068
test coverage
1. the degree to which a given test or set of tests addresses all specified requirements for a given system or component. ***2.*** extent to which the test cases test the requirements for the system or software product. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.51

3.3069
test criteria
1. the criteria that a system or component must meet in order to pass a given test

cf. acceptance criteria, pass/fail criteria

3.3070
test design
1. documentation specifying the details of the test approach for a software feature or combination of software features and identifying the associated tests. *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.32

NOTE commonly includes the organization of the tests into groups

3.3071
test design specification
1. a document specifying the details of the test approach for a software feature or combination of software features and identifying the associated tests

3.3072
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 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.7

NOTE Types include test case specification, test incident report, test log, test plan, test procedure, test report.

3.3073
test driver
1. a software module used to invoke a module under test and, often, provide test inputs, control and monitor execution, and report test results. *Syn:* test harness

3.3074
test effort
1. activity of performing one or more testing tasks. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.44

3.3075
test environment
1. hardware and software configuration necessary to conduct the test case. *ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.8

3.3076
test execution
1. act of performing one or more test cases

3.3077
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

3.3078
test incident report
1. document reporting on any event that occurs during the testing process which requires investigation

3.3079
test item
1. system or software item that is an object of testing*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.48

3.3080
test item transmittal report
1. document identifying test items

NOTE contains current status and location information

3.3081
test level
1. separate test effort that has its own documentation and resources*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.45

EXAMPLE component, component integration, system, and acceptance testing

3.3082
test log
1. chronological record of relevant details about the execution of tests

3.3083
test objective
1. identified set of software features to be measured under specified conditions by comparing actual behavior with the required behavior*. ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing.*4.9, *ISO/IEC 25062:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports.*4.9

3.3084
test phase
1. the 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

3.3085
test plan
1. a document describing the scope, approach, resources, and schedule of intended test activities*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.33. ***2.*** a document that describes the technical and management approach to be followed for testing a system or component*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.33. ***3.*** a plan that establishes detailed requirements, criteria, general methodology, responsibilities, and general planning for test and evaluation of a system*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.09

NOTE It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. Typical contents identify the items to be tested, tasks to be performed, responsibilities, schedules, and required resources for the testing activity.

3.3086
test procedure
1. detailed instructions for the setup, execution, and evaluation of results for a given test case*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.34. ***2.*** a document containing a set of associated instructions as in (1)*. IEEE Std 1012-2004 IEEE Standard for Software Verification and*

*Validation.*3.1.34. ***3.*** documentation that specifies a sequence of actions for the execution of a test. *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation.*3.1.34

3.3087
test procedure specification
1. document specifying a sequence of actions for the execution of a test

3.3088
test readiness review (TRR)
1. a 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.*** a review as in (1) for any hardware or software component

cf. code review, formal qualification review, design review, requirements review

3.3089
test repeatability
1. an attribute of a test, indicating that the same results are produced each time the test is conducted

3.3090
test report
1. a document that describes the conduct and results of the testing carried out for a system or component. *Syn:* test summary report

cf. test case specification, test incident report, test item transmittal report, test log, test plan, test procedure

3.3091
test set architecture
1. the nested relationships between sets of test cases that directly reflect the hierarchic decomposition of the test objectives. *IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

3.3092
test unit
1. a 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. *IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

3.3093
testability
1. extent to which an objective and feasible test can be designed to determine whether a requirement is met. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.52. ***2.*** the degree to which a requirement is stated in terms that permit establishment of test criteria and performance of tests to determine whether those criteria have been met. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.18. ***3.*** the degree to which a system can be unit tested and system tested. ***4.*** the effort required to test software. ***5.*** the 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

3.3094
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 Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.46

3.3095
testing description
1. description of the test execution conditions (i.e. test procedure). *ISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product 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

3.3096
testing task iteration
1. testing task that is re-performed during maintenance after having been originally performed during development. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.47

3.3097
testware
1. products produced by the testing effort. *IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation.*3.1.51

EXAMPLE documentation and data

3.3098
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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.03

EXAMPLE a screen; a business letter printed on paper or displayed on a screen

3.3099
text editor
1. a computer program, often part of a word processing system, that allows a user to enter, alter, and view text. *Syn:* editor

3.3100
text page
1. a model page that contains textual material related to a specific diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.2*.1.102

3.3101
text processing
1. data processing operations on text, such as entering, editing, merging, retrieving, storing, displaying, or printing. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.10. *Syn:* word processing

3.3102
think time
1. the 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

3.3103
third normal form
1. result of a normalization process that transforms groups of data so that each non-key attribute does not depend on any other non-key attribute. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.3104
thrashing
1. a 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

3.3105
threat
1. a state of the system or system environment which can lead to adverse effect in one or more given risk dimensions*. ISO/IEC 15026:1998, Information technology — System and software integrity levels*.3.21. ***2.*** a condition or situation unfavorable to the project, a negative set of circumstances, a negative set of events, a risk that will have a negative impact on a project objective if it occurs, or a possibility for negative changes*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3106
threat modeling
1. a 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, and/or denial of service*. IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process.*2.1.5

NOTE It results in a vulnerability assessment.

3.3107
three-address instruction
1. a 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

3.3108
three-plus-one address instruction
1. a 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

3.3109
three-point estimate
1. [Technique] an analytical technique that uses three cost or duration estimates to represent the optimistic, most likely, and pessimistic scenarios. This technique is applied to improve the accuracy of the estimates of cost or duration when the underlying activity or cost component is uncertain*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3110
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) — Fourth Edition*

3.3111
throughput
1. the amount of work that can be performed by a computer system or component in a given period of time ***2.*** the rate (i.e., the average number per time unit with respect to the rating interval) of all tasks of a task type submitted to the SUT*. ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.24

NOTE Usually throughput is defined by the rate of terminated tasks during a period of time.

3.3112
throughput rating value
1. the 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

3.3113
throughput reference value
1. the 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

3.3114
throwclear
1. foldout whose print area is such that all of the material on the page can be viewed with the book shut, so that it can be viewed at all times while looking at any of the preceding pages of the book. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.48

3.3115
time
1. in decreasing order of resolution, CPU execution time, elapsed time (i.e., wall clock time), or calendar time. *IEEE Std 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability.*2.6

3.3116
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) — Fourth Edition*

3.3117
time class
1. a 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 amount 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

3.3118
time out
1. a 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

3.3119
time sharing
1. a 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 Time sharing may be implemented by time slicing, priority-based interrupts, or other scheduling methods.

3.3120
time slicing
1. a 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.

3.3121
time-critical task
1. a task that must meet a hard deadline

3.3122
timeliness function
1. a description of the user requirements with respect to the execution times of tasks of a specific task type. *ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems.*4.29

NOTE It consists of one or more time classes.

3.3123
timeliness rating value
1. the 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

3.3124
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.*4.30

3.3125
timer event
1. a stimulus used to periodically activate a task

3.3126
time-scaled schedule network diagram
1. [Tool] 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) — Fourth Edition*

3.3127
timesharing
1. an operating technique of a data processing system that provides for the interleaving in time of two or more processes in one processor. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.43

3.3128
timing
1. the process of estimating or measuring the amount of execution time required for a software system or component

cf. sizing

3.3129
timing analyzer
1. a 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

3.3130
timing diagram
1. a diagram showing the time-ordered execution sequence of a group of tasks

3.3131
TINA
1. Telecommunication Information Networking Architecture*. ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.3132
tinderbox
1. an automated build and regression-testing tool

NOTE A tinderbox will typically fetch on a regular basis the latest versions of the software from each supported branch, build it for the different platforms, and report the results from the build and the regression tests.

3.3133
to-compete-performance index (TCPI)
1. the calculated projection of cost performance that must be achieved on the remaining work to meet a specified management goal, such as the budget at completion (BAC) or the estimate at completion (EAC). It is the ratio of 'remaining work' to the 'funds remaining.' *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3134
token
1. a 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

3.3135
tool
1. a 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) — Fourth Edition.* ***3.*** a 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 Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.16

NOTE particularly, but not exclusively, a modeling tool. Also, tool is used as a short form for software tool, and more specifically for CASE tool.

3.3136
top box
1. the box in the A-0 context diagram that models the top-level function of an IDEF0 model*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.103

3.3137
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 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.01.1

cf. bottom-up, critical piece first

EXAMPLE top-down design, top-down testing

3.3138
top-down design
1. a design approach in which a system's functionality is decomposed from high-level concepts into lower-level pieces. ***2.*** the 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. *Syn:* top-down decomposition

3.3139
topic
1. small part of a document that deals with a single subject*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.50

EXAMPLE instructions on how to print the current document

NOTE In printed documentation, a topic is equivalent to a section (heading; subheading) and its content. In onscreen documentation, a topic consists of a title (heading) and information about a subject (typically, a task or a concept or reference information). For on-screen documentation, the system may present a topic without user intervention.

3.3140
top-level function
1. the function modeled by the single box in the A-0 context diagram of an IDEF0 model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.104

3.3141
total
1. a complete mapping. *IEEE Std 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 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.

3.3142
total cluster
1. a subclass cluster in which each instance of a superclass must be an instance of at least one of the subclasses of the cluster*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.197. *Syn:* complete cluster

cf. incomplete cluster, partial cluster, superclass

3.3143
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

3.3144
total degree of influence (TDI)
1. the sum of the degrees of influence for the fourteen GSCs*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.3145
total float (TF)
1. the total amount of time that a schedule activity may be delayed from its early start date without delaying the project finish date, or violating a schedule constraint. Calculated using the critical path method technique and determining the difference between the early finish dates and late finish dates*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3146
total quality management (TQM)
1. a holistic approach to quality improvement in all life-cycle phases

3.3147
TP
1. Transaction Processing. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*4

3.3148
T-profile
1. Transfer profile. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.3149
TQM
1. Total Quality Management. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3150
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

3.3151
trace
1. a 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

NOTE Types include execution trace, retrospective trace, subroutine trace, symbolic trace, variable trace.

3.3152
traceability
1. the 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. *IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.19. ***2.*** the identification and documentation of derivation paths (upward) and allocation or flowdown paths (downward) of work products in the work product hierarchy. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology-System Definition -Concept of Operation Document.*3.24. ***3.*** the degree to which each element in a software development product establishes its reason for existing. ***4.*** discernable association among two or more logical entities, such as requirements, system elements, verifications, or tasks

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

3.3153
traceability matrix
1. a 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

3.3154
traceable
1. having components whose origin can be determined. *IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*H.3 f)

3.3155
trade secret
1. a formula, process, design, or intellectual property that is protected by non-disclosure

3.3156
trade study
1. evaluation of alternatives, based on criteria and systematic analysis, to select the best alternative for attaining determined objectives

3.3157
trademark
1. a symbol, word, or phrase used to denote a particular source of goods or services

3.3158
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 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.35

3.3159
trade-off analysis
1. an analytical evaluation of design options/alternatives against performance, design-to-cost objectives, and life cycle quality factors*. IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.*3.1.38

3.3160
trailer
1. Identification or control information placed at the end of a file or message

cf. header (2)

3.3161
trailing decision
1. a loop control that is executed after the loop body

cf. leading decision UNTIL

3.3162
training
1. provision of formal and informal learning activities

NOTE may include 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.

3.3163
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.*** an activity which leads to a set of object changes consistent with a dynamic schema (and its constraining invariant schema)*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*13.7.1.1

3.3164
transaction analysis
1. a software development technique in which the structure of a system is derived from analyzing the transactions that the system is required to process. *Syn:* transaction-centered design

cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transform analysis

3.3165
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 It is read one time only and its data is consumed.

3.3166
transaction matrix
1. a matrix that identifies possible requests for database access and relates each request to information categories or elements in the database

3.3167
transaction rate GSC
1. one of the 14 general system characteristics describing the degree to which the rate of business transactions influenced the development of the application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.3168
transaction schema
1. a dynamic schema and an invariant schema defining transactions and their dependencies*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*16.8.1.1

3.3169
transaction transparency
1. a distribution transparency which masks coordination of activities amongst a configuration of objects to achieve consistency*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*4.4.1.8

3.3170
transactional function
1. 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 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.

3.3171
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*

3.3172
transactional functions
1. the functionality provided to the user to process data by an application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE Transactional functions are defined as external inputs, external outputs, and external inquiries.

3.3173
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

3.3174
transfer file
1. a file containing data to be interchanged It is made up of a header, and a number of components*. ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.1

NOTE Components contain either data or data definition data.

3.3175
transform analysis
1. a 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. *Syn:* transformation analysis, transform-centered design

cf. data structure-centered design, input-process-output, modular decomposition, object-oriented design, rapid prototyping, stepwise refinement, structured design, transaction analysis

3.3176
transient error
1. an error that occurs once, or at unpredictable intervals

cf. intermittent fault, random failure

3.3177
transition
1. a 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. ***2.*** a 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

3.3178
transition condition
1. a 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

3.3179
transition mode
1. a 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

3.3180
transition occurrence
1. if a transition is enabled in a mode, it may 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. *Syn:* transition rule

NOTE 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 may be both an input place and an output place of the same transition.

3.3181
transition variables
1. the 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 These are the transition condition and the annotations of arcs surrounding the transition.

3.3182
translator
1. a 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

3.3183
trap
1. a 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)

3.3184
tree-structured chart
1. a 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

3.3185
trend
1. a time analysis showing repeated occurrences of a particular measure or metric. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

3.3186
trend analysis
1. [Technique] 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) — Fourth Edition*

3.3187
triggering event
1. an event that occurs outside the boundary of the measured software and initiates one or more functional processes. *ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.22

NOTE In a set of functional user requirements, each event-type which triggers a functional process is indivisible for that set of FUR. Clock and timing events can be triggering events. An event has either happened, or it has not, it is instantaneous.

3.3188
triggers
1. indications that a risk has occurred or is about to occur. Triggers may be discovered in the risk identification process and watched in the risk monitoring and control process. Triggers are sometimes called risk symptoms or warning signs. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3189
triple constraint
1. a framework for evaluating competing demands, such as schedule, cost, and quality

NOTE The triple constraint is often depicted as a triangle where one of the sides or one of the corners represents one of the parameters being managed by the project team.

3.3190
TRR
1. test readiness review

3.3191
trunk
1. the software's main line of development; the main starting point of most branches

NOTE One can often distinguish the trunk from other branches by the version numbers used for identifying its files, which are shorter than those of all other branches.

3.3192
tunnel notation
1. a 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 Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.107

3.3193
tunneled arrow
1. an arrow left undrawn between its attachment to an ancestral box and its appearance as a boundary arrow on some hierarchically consecutive descendent diagram. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.105

3.3194
tunneling
1. the act of applying tunnel notation to an arrow segment. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.106

3.3195
turnaround time
1. the 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

3.3196
turnkey
1. pertaining to a hardware or software system delivered in a complete, operational state

3.3197
turnkey system
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.48

NOTE Some preparatory work on the user's data may be required.

3.3198
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.51

3.3199
two-address instruction
1. a computer instruction that contains two address fields. *Syn:* double-operand instruction

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

3.3200
two-level address
1. an indirect address that specifies the storage location containing the address of the desired operand

cf. n-level address

3.3201
two-level encoding
1. a microprogramming technique in which different microoperations may be en coded 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. *Syn:* two level encoding

cf. bit steering, residual control, single-level encoding

3.3202
two-plus-one address instruction
1. a 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

3.3203
type
1. a set. *ISO/IEC 15909-1:2004, Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation.*2.1.27

3.3204
UDF
1. unit development folder

cf. software development file

3.3205
ULA
1. Upper Layers Architecture. *ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview*

3.3206
UML
1. Unified Modeling Language. *ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function.*4

3.3207
unadjusted function point count (UFP)
1. the measure of the functionality provided to the user by the project or application. *ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE It is contributed by the measure of two function types-data and transactional.

3.3208
unambiguous
1. described in terms that only allow a single interpretation, aided, if necessary, by a definition. *IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data.*H.3.a)

3.3209
unbundle
1. the separation of arrow meanings, expressed by branching arrow segments*. IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.108

NOTE that is, the separation of object types from an object type set

3.3210
uncertainty
1. the result of not having accurate or sufficient knowledge of a situation

NOTE often the root cause of a risk factor

3.3211
unconditional jump
1. a jump that takes place regardless of execution conditions

cf. conditional jump

3.3212
underflow exception
1. an 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

3.3213
underlying license
1. license for software use as originally purchased or procured, and which can typically be linked directly to purchase records*. ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes.*3. 15

NOTE An underlying license may have conditions associated with it, requiring it to be used in combination with another license or licenses to create an effective full license.

3.3214
understandability
1. the ease with which a system can be comprehended at both the system-organizational and detailed-statement levels

NOTE Understandability has to do with the system's coherence at a more general level than readability does.

3.3215
undirected graph
1. a graph (sense 2) in which no direction is implied in the internode connections

cf. directed graph

3.3216
Unified Modeling Language (UML)
1. a graphical language for visualizing, specifying, constructing, and documenting an object-oriented software-intensive system's artifacts

3.3217
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*

3.3218
uniqueness constraint
1. a constraint stating that no two distinct instances of a class may agree on the values of all the properties that are named in the uniqueness constraint. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.199

3.3219
unit
1. a separately testable element specified in the design of a computer software component. ***2.*** a logically separable part of a computer program. ***3.*** a software component that is not subdivided into other components. ***4.*** a quantity adopted as a standard of measurement. *ISO/IEC 14598-3:2000, Software engineering — Product evaluation — Part 3: Process for developers.*4.4

3.3220
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 magnitude relative to that quantity. *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.40; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.60

NOTE Units of measurement have conventionally assigned names and symbols.

3.3221
unit requirements documentation
1. documentation that sets forth the functional, interface, performance, and design constraint requirements for the test unit. *IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing.*2

3.3222
unit test
1. testing of individual routines and modules by the developer or an independent tester. ***2.*** a test of individual programs or modules in order to ensure that there are no analysis or programming errors. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.05. ***3.*** test of individual hardware or software units or groups of related units

3.3223
unit test framework
1. an environment that facilitates unit testing

3.3224
unpack
1. to recover the original form of one or more data items from packed data

cf. pack

3.3225
unspecialize
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.200

cf. respecialize, specialize

3.3226
unstratified language
1. a language that can be used as its own metalanguage

EXAMPLE English or German

3.3227
UNTIL
1. a single-entry, single-exit loop, in which the loop control is executed after the loop body. *Syn:* post-tested iteration

cf. closed loop, WHILE, trailing decision

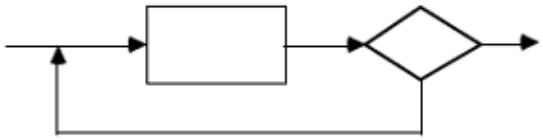


Figure 18 — UNTIL

3.3228
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

3.3229
up
1. pertaining to a system or component that is operational and in service

cf. down busy, idle

NOTE Such a system is either busy or idle.

3.3230
up time
1. the 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

3.3231
updatable argument
1. the designation given to an operation argument that identifies an instance to which a request may be sent that will change the state of the instance*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.201

NOTE An argument not designated as ""updatable"" means that there will be no requests sent that will change the state of the instance identified by the argument.

3.3232
upload
1. to transfer programs or data from a connected computer to a computer with greater resources*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.37

NOTE typically from a personal computer to a server

3.3233
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

3.3234
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

3.3235
UR
1. user requirements*. ISO/IEC TR 14143-4:2002, Information technology — Software measurement — Functional size measurement — Part 4: Reference model.*4

3.3236
usability
1. a measure of an executable software unit's or system's functionality, ease of use, and efficiency*. IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes.* 3.21. ***2.*** the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component. ***3.*** the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. *ISO/IEC 25062:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports.* 4.10. ***4.*** capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions. *ISO/IEC 9126-1:2001 Software engineering — Product quality — Part 1: Quality model.* 6.3.

cf. reusability

NOTE This term has been deliberately redefined to more properly convey its meaning in the software reuse context.

3.3237
usability laboratory
1. typically, a suite of evaluation and observation rooms which may be fitted with video and audio equipment for recording user responses*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.49

3.3238
usability test
1. a test to determine whether an implemented system fulfils its functional purpose as determined by its users*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.08. *Syn:* fitness-for-use test

3.3239
usability testing
1. formal process for evaluating the suitability of documentation*. ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.50

3.3240
usage mode
1. primary manner in which the documentation developer expects the document to be used*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.53

3.3241
use case
1. in UML, a complete task of a system that provides a measurable result of value for an actor

NOTE More formally, a use case defines a set of use case instances or scenarios.

3.3242
use case diagram
1. a UML diagram that shows actors, use cases, and their relationships

3.3243
use case model
1. a model that describes a system's functional requirements in terms of use cases

3.3244
use case specification
1. a document that describes a use case

NOTE A use case specification's fundamental parts are the use case name, brief description, precondition, basic flow, postcondition, and alternate flow.

3.3245
use of IT
1. planning, design, development, deployment, operation, management, and application of IT to meet the needs of the business*. ISO/IEC 38500:2008, Corporate governance of information technology.*1.6.18

NOTE includes both the demand for and the supply of IT services by internal business units, specialist IT units, or external suppliers and utility services (such as those providing software as services)

3.3246
user
1. person who performs one or more tasks with software; a member of a specific audience*. ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.54. ***2.*** person who interacts with the product*. ISO/IEC 25062:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Common Industry Format (CIF) for usability test reports.*4.6. ***3.*** individual or organization who uses a software-intensive system in daily work activities or recreational pursuits*. IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.25. ***4.*** individual or group that benefits from a system during its utilization*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*3.41; *ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.41. ***5.*** 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.*3.11; *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.9. ***6.*** a 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.*** the person who derives engineering value through interaction with a CASE tool*. IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections.*3.17

cf. developer

NOTE The user may perform other roles such as acquirer or maintainer. The role of user and the role of operator may be vested, simultaneously or sequentially, in the same individual or organization.

3.3247
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.55

cf. user manual

3.3248
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

3.3249
user interface
1. an interface that enables information to be passed between a human user and hardware or software components of a computer system ***2.*** ensemble of software and hardware that allows a user to interact with a computer system. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.56

3.3250
user interface task
1. a task that hides the details of the interface to and interacts sequentially with a human user

3.3251
user manual
1. a document that presents the information necessary to employ a system or component to obtain desired results. ***2.*** a document that describes how to use a functional unit, and that may include description of the rights and responsibilities of the user, the owner, and the supplier of the unit. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.06.03. *Syn:* user guide, user's manual, users' manual

cf. data input sheet, diagnostic manual, installation manual, operator manual, programmer manual, support manual

NOTE Typically described are system or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions. A user manual is distinguished from an operator manual when a distinction is made between those who operate a computer system (mounting tapes, etc.)and those who use the system for its intended purpose.

3.3252
user need
1. a user requirement for a system that a user believes would solve a problem experienced by the user. *IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document.*3.26. ***2.*** set of functional user requirements and non-functional user requirements that the users need the system to fulfill. *ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*A.17

3.3253
user requirements (UR)
1. the requirements of the system's customers or end-users. *ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK).*2.1.6. ***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

NOTE User requirements comprise two subsets: functional user requirements and non-functional user requirements.

3.3254
user terminal
1. a terminal that enables a user to communicate with a computer. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.12

3.3255
user type
1. a 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

3.3256
user view
1. a formal representation of the user's business needs in the user's language*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***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 Developers translate the user information into information technology language in order to provide a solution.

3.3257
user-friendly
1. pertaining to ease and convenience of use by humans*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.06.22. ***2.*** pertaining to a computer system, device, program, or document designed with ease of use as a primary objective

3.3258
user-identifiable
1. defined requirements for processes and/or groups of data that are agreed upon, and understood by, both the user(s) and software developer(s)*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual.*6

3.3259
utility
1. a software tool designed to perform some frequently used support function. ***2.*** a measure of value within a given value system, often measured on a scale of 0 to 100

EXAMPLE a program to copy magnetic tapes

3.3260
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

3.3261
utilization bound theorem
1. a 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

3.3262
V&V
1. verification and validation

3.3263
validated metric
1. a metric whose values have been statistically associated with corresponding quality factor values*. IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology.*2.25

3.3264
validation
1. confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled*. ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.37. ***2.*** the process of providing evidence that the software and its associated products satisfy system requirements allocated to software at the end of each life cycle activity, solve the right problem, and satisfy intended use and user needs*. IEEE Std 1012-2004 IEEE Standard for 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*

*(IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.54. ***4.*** the process of evaluating a system or component during or at the end of the development process to determine whether a system or component satisfies specified requirements*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.20 ***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) — Fourth Edition*

cf. verification

NOTE Validation demonstrates that the system can be used by the users for their specific tasks. "Validated" is used to designate the corresponding status. [ISO 9000:2005] In design and development, validation concerns the process of examining a product to determine conformity with user needs. Validation is normally performed on the final product under defined operating conditions. It may be necessary in earlier stages. Multiple validations may be carried out if there are different intended uses.

3.3265
validation test
1. a test to determine whether an implemented system fulfils its specified requirements*. ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.04

3.3266
value
1. number or category assigned to an attribute of an entity by making a measurement*. ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.63. ***2.*** numerical or categorical result assigned to a base measure, derived measure, or indicator*. ISO/IEC 15939:2007, Systems and software engineering — Measurement process.*3.42. ***3.*** an entity that may be a possible actual parameter in a request*. ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.2.26. ***4.*** magnitude of a particular quantity, generally expressed as a unit of measurement multiplied by a number*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.3

3.3267
value adjustment factor (VAF)
1. the factor that indicates the general functionality provided to the user of the application*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE The VAF is calculated based on an assessment of the 14 general system characteristics (GSCs) for an application.

3.3268
value class
1. a class that represents instances that are pure values*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.202

NOTE The constituent instances of a value class do not come and go and cannot change state.

3.3269
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) — Fourth Edition*

3.3270
value list constraint
1. a constraint that specifies the set of all acceptable instance values for a value class*. IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.203

3.3271
value range constraint
1. a 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 Std 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 A range constraint only makes sense if there is a linear ordering specified.

3.3272
variable
1. a quantity or data item whose value can change. ***2.*** an instance whose identity is unknown at the time of writing. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.205. ***3.*** a data item whose value can change during program execution

NOTE A variable is represented by an identifier that begins with an upper-case letter.

3.3273
variable trace
1. a record of the name and values of variables accessed or changed during the execution of a computer program. *Syn:* data-flow trace, data trace, value trace

cf. execution trace, retrospective trace, subroutine trace, symbolic trace

3.3274
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) — Fourth Edition*

3.3275
variance analysis
1. [Technique] a method for resolving the total variance in the set of scope, cost, and schedule variables into specific component variances that are associated with defined factors affecting the scope, cost, and schedule variables. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3276
variant
1. in fault tolerance, a version of a program resulting from the application of software diversity

3.3277
VDD
1. version description document

3.3278
vendor branch
1. a branch for keeping track of versions of imported software

NOTE Differences between successive versions can then be readily applied to the locally modified import.

3.3279
Ver
1. version. *ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method*.4

3.3280
verb phrase
1. a 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 Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.206. ***2.*** a phrase used to name a relationship, which consists of a verb and words that constitute the object of the phrase*. IEEE Std 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 A verb phrase is ideally stated in active voice.

3.3281
verifiable
1. can be checked for correctness by a person or tool*. IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data*.H.3 c)

3.3282
verification
1. the process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*.3.1.36. ***2.*** formal proof of program correctness. ***3.*** confirmation, through the provision of objective evidence, that specified requirements have been fulfilled*. ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.55; *ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes.*4.38; *ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE.*4.64. ***4.*** confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled*. ISO/IEC 9126-1:2001, Software engineering — Product quality — Part 1: Quality model*. ***5.*** 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) — Fourth Edition*. ***6.*** process of providing objective evidence that the software and its associated products comply with 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 (e.g., building the software correctly)*. IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation*.3.1.54

cf. validation

NOTE [ISO 9000:2005] "Verified" is used to designate the corresponding status. In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity. A system may be verified to meet the stated requirements, yet be unsuitable for operation by the actual users.

3.3283
verification and validation (V&V)
1. the process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements

cf. independent verification and validation

3.3284
verification and validation (V&V) effort
1. the work associated with performing the V&V processes, activities, and tasks*. IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*.3.1.37

3.3285
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.*3.10

NOTE The purpose of applying this part of ISO/IEC 14143 is to enable the user to select the FSM method which "best" meets their needs. Therefore, 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.

3.3286
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

3.3287
verification test
1. a test of a system to prove that it meets all its specified requirements at a particular stage of its development. *ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development.*20.05.03

3.3288
verify scope
1. [Process] the process of formalizing acceptance of the completed project deliverables. *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3289
version
1. an initial release or re-release of a computer software configuration item, associated with a complete compilation or recompilation of the computer software configuration item ***2.*** an initial release or complete re-release of a document, as opposed to a revision resulting from issuing change pages to a previous release ***3.*** an operational software product that differs from similar products in terms of capability, environmental requirements, and configuration ***4.*** an identifiable instance of a specific file or release of a complete system. ***5.*** identified instance of an item. *ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes.*4.56

NOTE Modification to a version of a software product resulting in a new version requires configuration management action.

3.3290
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

3.3291
version description document (VDD)
1. a document that accompanies and identifies a given version of a system or component

NOTE Typical contents include an inventory of system or component parts, identification of changes incorporated into this version, and installation and operating information unique to the version described.

3.3292
verso
1. page on the opposite side (i.e. right or left) from the front cover. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.53

3.3293
vertical microinstruction
1. a microinstruction that specifies one of a sequence of operations needed to carry out a machine language instruction

cf. diagonal microinstruction, horizontal microinstruction

NOTE Vertical microinstructions are relatively short, 12 to 24 bits, and are called 'vertical' because a sequence of such instruction, normally listed vertically on a page, is required to carry out a single machine language instruction.

3.3294
video display terminal (VDT)
1. a user terminal with a display screen and usually equipped with an input unit such as a keyboard. *ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.16. *Syn:* visual display terminal, visual display unit (VDU)

3.3295
view
1. a developer's copy of a branch. ***2.*** a collection of subject domains, classes, relationships, responsibilities, properties, constraints, and notes assembled or created for a certain purpose and covering a certain scope. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.207. ***3.*** a collection of entities and assigned attributes (domains) assembled for some purpose. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.207. ***4.*** a set of related categories. *ISO/IEC TR 12182:1998, Information technology — Categorization of software.*4.2. ***5.*** a representation of a whole system from the perspective of a related set of concerns. *IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*3.9

NOTE A view may cover the entire area being modeled or only a part of that area.

3.3296
view diagram
1. a graphic representation of the underlying semantics of a view. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.208

3.3297
viewpoint
1. a specification of the conventions for constructing and using a view. *IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.*3.1

NOTE 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.

3.3298
viewpoint (on a system)
1. a form of abstraction achieved using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within a system. *ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations.*3.2.7

3.3299
viewpoint statement
1. a brief statement of the perspective of an IDEF0 model that is presented in the a-0 context diagram of the model. *IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0.*2.1.109

3.3300
violation
1. a behavior contrary to that required by a rule*. ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language.*6.4.3

NOTE A rule or policy may provide behavior to occur upon violation of that or some other rule or policy.

3.3301
virtual
1. pertaining to a functional unit that appears to be real, but whose functions are accomplished by other means*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.49. ***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

3.3302
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

3.3303
virtual machine (VM)
1. a 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 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.01.50

3.3304
virtual reference
1. references made to concepts other than specific meta-entities in a metamodel*. ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE represented by boxes with diagonal striping

3.3305
virtual storage
1. a storage allocation technique in which auxiliary storage can be addressed as though it were part of main storage. *Syn:* multilevel storage, virtual memory

cf. real storage, virtual address, paging (2)

NOTE Portions of a user's program and data are placed in auxiliary storage, and the operating system automatically swaps them in and out of main storage as needed.

3.3306
virtual team
1. a group of persons with a shared objective who fulfill their roles with little or no time spent meeting face to face. Various forms of technology are often used to facilitate communication among team members. Virtual teams can be comprised of persons separated by great distances*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3307
visibility
1. the degree to which a transaction can access object state concurrently with other transactions*. ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture.*13.7.1.3. ***2.*** the specification, for a property, of "who can see it?" *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)*.3.1.209

NOTE that is, whose methods can reference the property. Visibility is private, protected, or public.

3.3308
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) — Fourth Edition*

3.3309
VSCID
1. vender service context codeset ID. *ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP).*3.3

3.3310
waiver
1. a 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

3.3311
walk-through
1. a 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

3.3312
war room
1. a room used for project conferences and planning, often displaying charts of cost, schedule status, and other key project data

3.3313
warning
1. advisory information in documentation that states that performing some action may lead to serious or dangerous consequences. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.57

cf. caution, note

3.3314
waterfall model
1. a 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

3.3315
WBS
1. work breakdown structure. *ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management.*5

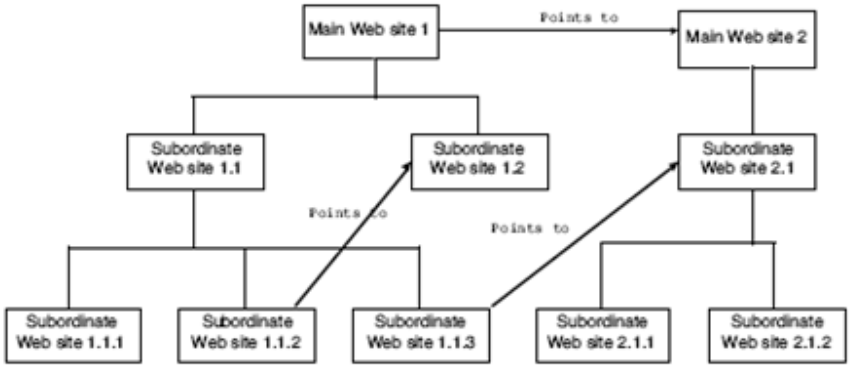
3.3316
wearout-failure period
1. the period in the life cycle of a system or component during which hardware failures occur at an increasing rate due to deterioration

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

3.3317
web page
1. a digital multimedia object as delivered to a client system*. IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.11

NOTE A web page may be generated dynamically from the server side, and may incorporate applets or other elements active on either the client or server side.

3.3318
web site
1. a collection of logically connected web pages managed as a single entity. *IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle*.3.1.12



NOTE—All elements of this diagram may represent separately managed Web sites. Such management should reflect deference to the applicable policies of the organization hierarchy. Note that Web sites are not implicitly hierarchical, it is the organization hierarchy policies that may be relevant.

Figure 19 — Web site

NOTE A web site may contain one or more subordinate web sites.

3.3319
well-engineered web site
1. a web site designed and implemented in accordance with the recommendations of IEEE Std 2001-2002*. IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle.*3.1.13

NOTE Frame, NoFrame and Robot are used based on the HTML 4.01 specification.

3.3320
well-formed requirement
1. a statement of system functionality (a capability) that can be validated, and that must be met or possessed by a system to solve a customer problem or to achieve a customer objective, and is qualified by measurable conditions and bounded by constraints*. IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications.*3.22

3.3321
WHILE
1. a single-entry, single-exit loop in which the loop control is executed before the loop body. *Syn:* pretested iteration

cf. closed loop, UNTIL, leading decision

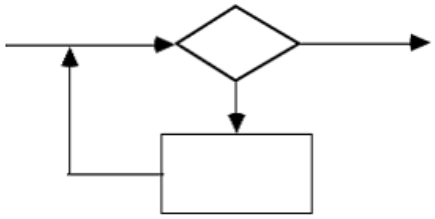


Figure 20 — WHILE

3.3322
whitespace
1. the nondisplaying formatting characters that are embedded within a block of free text. *IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject).*3.1.210

EXAMPLE spaces and tabs

3.3323
widow
1. line of text on its own at the start of a page. *ISO/IEC 15910:1999, Information technology — Software user documentation process.*4.56

3.3324
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, Systems and software engineering — Requirements for designers and developers of user documentation.*4.58

3.3325
wizard
1. procedural form of help that guides a user through each step of a task through dialog with the user. *ISO/IEC 26514, Systems and software engineering — Requirements for designers and developers of user documentation.*4.59

3.3326
word
1. a sequence of bits or characters that is stored, addressed, transmitted, and operated on as a unit within a given computer. ***2.*** an element of computer storage that can hold a sequence of bits or characters as in (1). ***3.*** a sequence of bits or characters that has meaning and is considered an entity in some language

cf. computer word

3.3327
work activity
1. a collection of work tasks spanning a fixed duration within the schedule of a software project. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.9

NOTE Work activities may contain other work activities, as in a work breakdown structure. The lowest-level work activities in a hierarchy of activities are work tasks. Typical work activities include project planning, requirements specification, software design, implementation, and testing.

3.3328
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) — Fourth Edition*

3.3329
work authorization system
1. [Tool] 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) — Fourth Edition*

3.3330
work breakdown structure (WBS)
1. [Output/Input] a deliverable-oriented hierarchical decomposition of the work to be executed by the project team to accomplish the project objectives and create the required deliverables. It organizes and defines the total scope of the project*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3331
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) — Fourth Edition*

3.3332
work breakdown structure dictionary
1. [Output/Input] a document that describes each component in the work breakdown structure (WBS). For each WBS component, the WBS dictionary includes a brief definition of the scope or statement of work, defined deliverable(s), a list of associated activities, and a list of milestones. Other information may include: responsible organization, start and end dates, resources required, an estimate of cost, charge number, contract information, quality requirements, and technical references to facilitate performance of the work*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3333
work effort
1. labor resources required for the production of a specified output*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual*

NOTE Here referring to the effort required to develop or maintain an application. Labor resources are usually expressed as work hours.

3.3334
work package
1. specification of the work that must be accomplished to complete a work task. *IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.1. ***2.*** a deliverable or project work component at the lowest level of each branch of the work breakdown structure*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

NOTE A work package should have a unique name and identifier, preconditions for initiating the work, staffing requirements, other needed resources, work products to be generated, estimated duration, risks factors, predecessor and successor work tasks, any special considerations for the work, and the completion criteria for the work package-including quality criteria for the work products to be generated.

3.3335
work performance information
1. [Output/Input] information and data, on the status of the project schedule activities being performed to accomplish the project work, collected as part of the direct and manage project execution processes. Information includes: status of deliverables; implementation status for change requests, corrective actions, preventive actions, and defect repairs; forecasted estimates to complete; reported percent of work physically completed; achieved value of technical performance measures; start and finish dates of schedule activities*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3336
work product
1. an artifact associated with the execution of a process*. ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary.*3.55. ***2.*** the product that is created by information systems work, here the result of a software development effort*. ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual* ***3.*** a tangible item produced during the process of developing or modifying software*. IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.1

EXAMPLE the project plan, supporting process requirements, design documentation, source code, test plans, meeting minutes, schedules, budgets, and problem reports

NOTE There are four generic product categories, as follows: services (e.g., operation); software (e.g., computer program, documents, information, contents); hardware (e.g., computer, device); processed materials. Some subset of the work products will be baselined and some will form the set of project deliverables.

3.3337
work task
1. the smallest unit of work subject to management accountability*. IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans.*3.12

NOTE A work task must be small enough to allow adequate planning and control of a software project, but large enough to avoid micro-management. The specification of work to be accomplished in completing a work task should be documented in a work package. Related work tasks should be grouped to form supporting processes and work activities.

3.3338
workaround
1. [Technique] a response to a negative risk that has occurred. Distinguished from contingency plan in that a workaround is not planned in advance of the occurrence of the risk event*. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition*

3.3339
working metamodel
1. the definition of the specific meta-objects that may be instantiated in the model section of a CDIF transfer*. ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview.*4.2

NOTE The working metamodel comprises the meta-objects in the CDIF semantic metamodel that are used by the subject areas referenced in the metamodel section of the transfer, and the meta-objects defined as extensions in the metamodel section.

3.3340
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

3.3341
working space
1. that portion of main storage that is assigned to a computer program for temporary storage of data. *Syn:* working area, working storage

3.3342
workload
1. the mix of tasks typically run on a given computer system

NOTE Major characteristics include input/output requirements, amount and kinds of computation, and computer resources required.

3.3343
workload model
1. a model used in computer performance evaluation, depicting resource utilization and performance measures for anticipated or actual workloads in a computer system

cf. system model

3.3344
workstation
1. a functional unit that usually has special purpose computing capabilities and includes user-oriented input units and output units*. ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms.*01.03.13

EXAMPLE a programmable terminal, a nonprogrammable terminal or a standalone microcomputer

3.3345
write
1. to record data in a storage device or on a data medium

cf. read

3.3346
write (-type)
1. a data movement type that moves a data group lying inside the functional process to persistent storage*. ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method.*3.26

NOTE A Write is considered to include certain associated data manipulations necessary to achieve the Write.

3.3347
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 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method.*3.1

NOTE The number of writing references is greater than 0 with all BFC types where it is applicable.

3.3348
XFN
1. X/Open Federated Naming*. ISO/IEC 14771:1999, Information technology — Open Distributed Processing — Naming framework.*4

3.3349
zero-address instruction
1. a 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 Source Standards

[1] A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Fourth Edition

[2] IEEE Std 1008-1987 (R1993, R2002) IEEE Standard for Software Unit Testing

[3] IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation

[4] IEEE Std 1016-1998 IEEE Recommended Practice for Software Design Descriptions

[5] IEEE Std 1028-2008 IEEE Standard for Software Reviews and Audits

[6] IEEE Std 1044-1993 (R2002) IEEE Standard Classification for Anomalies

[7] IEEE Std 1058-1998 IEEE Standard for Software Project Management Plans

[8] IEEE Std 1061-1998 (R2004) IEEE Standard for Software Quality Metrics Methodology

[9] IEEE Std 1062, 1998 Edition (R2002) IEEE Recommended Practice for Software Acquisition (includes IEEE Std 1062a)

[10] IEEE Std 1063-2001 (R2007) IEEE Standard for Software User Documentation

[11] IEEE Std 1074-2006 IEEE Standard for Developing a Software Project Life Cycle Process

[12] IEEE Std 1175.1-2002 (R2007) IEEE Guide for CASE Tool Interconnection — Classification and Description

[13] IEEE Std 1175.2-2006 IEEE Recommended Practice for CASE Tool Interconnection — Characterization of Interconnections

[14] IEEE Std 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process

[15] IEEE/EIA 12207.1-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Life cycle data

[16] IEEE/EIA 12207.2-1997 IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995, Standard for Information Technology — Software Life Cycle Processes — Implementation considerations

[17] IEEE Std 1228-1994 (R2002) IEEE Standard for Software Safety Plans

[18] IEEE Std 1233-1998 (R2002) IEEE Guide for Developing System Requirements Specifications

[19] IEEE Std 1320.1-1998 (R2004) IEEE Standard for Functional Modeling Language — Syntax and Semantics for IDEF0

[20] IEEE Std 1320.2-1998 (R2004) IEEE Standard for Conceptual Modeling Language — Syntax and Semantics for IDEF1X97 (IDEFobject)

[21] IEEE Std 1362-1998 (R2007) IEEE Guide for Information Technology — System Definition — Concept of Operation Document

[22] IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software — Intensive Systems

[23] ISO/IEC 14764:2006 (IEEE Std 14764-2006), Software Engineering — Software Life Cycle Processes — Maintenance

[24] IEEE Std 1517-1999 (R2004) IEEE Standard for Information Technology — Software Life Cycle Processes — Reuse Processes

[25] IEEE Std 828-2005 IEEE Standard for Software Configuration Management Plans

[26] IEEE Std 829-2008 IEEE Standard for Software and System Test Documentation

[27] IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications

[28] IEEE Std 982.1-2005 IEEE Standard Dictionary of Measures of the Software Aspects of Dependability

[29] IEEE Std 2001-2002 IEEE Recommended Practice for the Internet — Web Site Engineering, Web Site Management, and Web Site Life Cycle

[30] ISO 3535:1977, Forms design sheet and layout chart

[31] ISO 5806:1984, Information processing — Specification of single-hit decision tables

[32] ISO 5807:1985, Information processing — Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts

[33] ISO 6593:1985, Information processing — Program flow for processing sequential files in terms of record groups

[34] ISO 9127:1988, Information processing systems — User documentation and cover information for consumer software packages

[35] ISO/IEC 10746-1:1998, Information technology — Open Distributed Processing — Reference model: Overview

[36] ISO/IEC 10746-2:1996, Information technology — Open Distributed Processing — Reference Model: Foundations

[37] ISO/IEC 10746-3:1996, Information technology — Open Distributed Processing — Reference Model: Architecture

[38] ISO/IEC 11411:1995, Information technology — Representation for human communication of state transition of software

[39] ISO/IEC 12207:2008 (IEEE Std 12207-2008), Systems and software engineering — Software life cycle processes

[40] ISO/IEC 13235-3:1998, Information technology — Open Distributed Processing — Trading Function — Part 3: Provision of Trading Function using OSI Directory service

[41] ISO/IEC 14102:2008, Information Technology — Guideline for the evaluation and selection of CASE tools

[42] ISO/IEC 14143-1:2007, Information technology — Software measurement — Functional size measurement; Part 1: Definition of concepts

[43] ISO/IEC 14143-2:2002, Information technology — Software measurement — Functional size measurement — Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998

[44] ISO/IEC 14143-6, Information technology — Software measurement — Functional size measurement — Part 6: Guide for use of ISO/IEC 14143 series and related International Standards

[45] ISO/IEC 14568:1997, Information technology — DXL: Diagram eXchange Language for tree-structured charts

[46] ISO/IEC 14598-1:1999, Information technology — Software product evaluation — Part 1: General overview

[47] ISO/IEC 14598-3:2000, Software engineering — Product evaluation — Part 3: Process for developers

[48] ISO/IEC 14598-4:1999, Software engineering — Product evaluation — Part 4: Process for acquirers

[49] ISO/IEC 14598-5:1998, Information technology — Software product evaluation — Part 5: Process for evaluators

[50] ISO/IEC 14752:2000, Information technology — Open Distributed Processing — Protocol support for computational interactions

[51] ISO/IEC 14753:1999, Information technology — Open Distributed Processing — Interface references and binding

[52] ISO/IEC 14756:1999, Information technology — Measurement and rating of performance of computer-based software systems

[53] ISO/IEC 14769:2001, Information technology — Open Distributed Processing — Type Repository Function

[54] ISO/IEC 14771:1999, Information technology — Open Distributed Processing — Naming framework

[55] ISO/IEC 15026:1998, Information technology — System and software integrity levels

[56] ISO/IEC 15288:2008 (IEEE Std 15288-2008), Systems and software engineering — System life cycle processes

[57] ISO/IEC 15289:2006, Systems and software engineering — Content of systems and software life cycle process information products (Documentation)

[58] ISO/IEC 15414:2006, Information technology — Open distributed processing — Reference model — Enterprise language

[59] ISO/IEC 15474-1:2002, Information technology — CDIF framework — Part 1: Overview

[60] ISO/IEC 15474-2:2002, Information technology — CDIF framework — Part 2: Modelling and extensibility

[61] ISO/IEC 15475-2:2002, Information technology — CDIF transfer format — Part 2: Syntax SYNTAX.1

[62] ISO/IEC 15475-3:2002, Information technology — CDIF transfer format — Part 3: Encoding ENCODING.1

[63] ISO/IEC 15476-4:2005, Information technology — CDIF semantic metamodel — Part 4: Data models

[64] ISO/IEC 15504-1:2004, Information technology — Process assessment — Part 1: Concepts and vocabulary

[65] ISO/IEC 15909-1:2004, Software and system engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation

[66] ISO/IEC 15910:1999, Information technology — Software user documentation process

[67] ISO/IEC 15939:2007, Systems and software engineering — Measurement process

[68] ISO/IEC 15940:2006, Information Technology — Software Engineering Environment Services

[69] ISO/IEC 16085:2006 (IEEE Std 16085-2006), Systems and software engineering — Life cycle processes — Risk management

[70] ISO/IEC 19500-2:2003, Information technology — Open Distributed Processing — Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)

[71] ISO/IEC 19761:2003, Software engineering — COSMIC-FFP — A functional size measurement method

[72] ISO/IEC 19770-1:2006, Information technology — Software asset management — Part 1: Processes

[73] ISO/IEC 20000-1:2005, Information technology — Service management — Part 1: Specification

[74] ISO/IEC 20926:2003, Software engineering — IFPUG 4.1 Unadjusted functional size measurement method — Counting practices manual

[75] ISO/IEC 20968:2002, Software engineering — Mk II Function Point Analysis — Counting Practices Manual

[76] ISO/IEC 2382-1:1993, Information technology — Vocabulary — Part 1: Fundamental terms

[77] ISO/IEC 2382-20:1990, Information technology — Vocabulary — Part 20: System development

[78] 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

[79] ISO/IEC 24744:2007, Software Engineering — Metamodel for Development Methodologies

[80] ISO/IEC 25000:2005, Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE

[81] ISO/IEC 25001:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Planning and management

[82] ISO/IEC 25020:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Measurement reference model and guide

[83] IISO/IEC 25051:2006, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Requirements for quality of Commercial Off-The-Shelf (COTS) software product 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 26514, Systems and software engineering — Requirements for designers and developers of user documentation

[86] ISO/IEC 29881:2008, Information technology — Software and systems engineering — FiSMA 1.1 functional size measurement method

[87] ISO/IEC 38500:2008, Corporate governance of information technology

[88] ISO/IEC 8631:1989, Information technology — Program constructs and conventions for their representation

[89] ISO/IEC 90003:2004, Software engineering — Guidelines for the application of ISO 9001:2000 to computer software

[90] ISO/IEC 9126-1:2001, Software engineering — Product quality — Part 1: Quality model

[91] ISO/IEC TR 12182:1998, Information technology — Categorization of software

[92] ISO/IEC TR 14143-3:2003, Information technology — Software measurement — Functional size measurement — Part 3: Verification of functional size measurement methods

[93] ISO/IEC TR 14143-4:2002, Information technology — Software measurement — Functional size measurement — Part 4: Reference model

[94] 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

[95] ISO/IEC TR 14471:2007, Information technology — Software engineering — Guidelines for the adoption of CASE tools

[96] ISO/IEC TR 14759:1999, Software engineering — Mock up and prototype — A categorization of software mock up and prototype models and their use

[97] ISO/IEC TR 15846:1998, Information technology — Software life cycle processes — Configuration Management

[98] ISO/IEC TR 16326:1999, Software engineering — Guide for the application of ISO/IEC 12207 to project management

[99] ISO/IEC TR 19759:2005, Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK)

[100] ISO/IEC TR 25021:2007, Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Quality measure elements

[101] ISO/IEC TR 9126-3:2003, Software engineering — Product quality — Part 3: Internal metrics

[102] ISO/IEC TR 9126-4:2004, Software engineering — Product quality — Part 4: Quality in use metrics

[103] ISO/IEC TR 9294:2005, Information technology — Guidelines for the management of software documentation

[104] ISO/IEC TR90005:2008, Systems engineering — Guidelines for the application of ISO 9001 to system life cycle processes

Annex B
(informative)

List of References

The systems and software engineering standards within the scope of ISO JTC 1/SC 7 refer to definitions from other standards maintained by other ISO Technical Committees and liaison organizations. These are listed here as references for ISO/IEC 24765.

[1] The IEEE Standards Dictionary: Glossary of Terms & Definitions

[2] ISO/IEC Guide 2:2004, Standardization and related activities — General vocabulary

[3] ISO/IEC Guide 51:1999, Safety aspects — Guidelines for their inclusion in standards

[4] ISO/IEC Guide 73:2002, Risk management — Vocabulary — Guidelines for use in standards

[5] ISO/IEC Guide 99:2007, International vocabulary of metrology — Basic and general concepts and associated terms (VIM)

[6] ISO 10241:1992, International terminology standards — Preparation and layout

[7] ISO 1087: 1990, Terminology — Vocabulary

[8] ISO 216:2007, Writing paper and certain classes of printed matter — Trimmed sizes — A and B series, and indication of machine direction

[9] ISO 20016:1975 Writing paper and certain classes of printed matter — Trimmed sizes — A and B series

[10] ISO 9000-2005, Quality management systems — Fundamentals and vocabulary

[11] ISO 9241-1:1997, Ergonomic requirements for office work with visual display terminals (VDTs) — Part 1: General introduction

[12] ISO 9241-11:1998, Ergonomic requirements for office work with visual display terminals (VDTs) — Part 11: Guidance on usability

[13] ISO/IEC 10181-2 [ITU-T Rec. X. 81 1 II], Information technology — Open Systems Interconnection — Security frameworks for open systems: Authentication framework

[14] ISO/IEC 10181-3 [ITU-T Rec. X. 812 I], Information technology — Open Systems Interconnection — Security frameworks for open systems: Access control framework

[15] ISO/IEC 10181-4 [ITU-T Ret X. 8 13 I], Information technology — Open Systems Interconnection — Security frameworks for open systems: Non-repudiation framework

[16] ISO/IEC 10181-5 [ITU-T Rec. X. 814 I], Information technology — Open Systems Interconnection — Security frameworks for open systems: Confidentiality framework

[17] ISO/IEC 10181-6 [ITU-T Rec. X. 815], Information technology — Open Systems Interconnection — Security frameworks for open systems: Integrity framework

[18] ISO/IEC 10181-7 [ITU-T Rec. X. 816 1,] Information technology — Open Systems Interconnection — Security frameworks for open systems: Security audit and alarms framework

[19] ISO/IEC 11770-1, Information technology — Security techniques — Key management — Part 1: Framework

[20] ISO/IEC 13235-1 [ITU-T Rec. X. 950], Information technology — Open Distributed Processing — Trading Function: Specification

[21] ISO/IEC 14753 [ITU-T Rec. X. 930], Information technology — Open Distributed Processing — Interface References and Binding)

[22] ISO/IEC 7498-1 [ITU-T Rec. X. 200-1], Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model

[23] ISO/IEC 9646-1, Information technology — Open Systems Interconnection — Conformance testing methodology and framework — Part 1: General concepts

[24] ITU-T Recommendation Z. 100 Specification and Description Language (SDL)

IEEE Notice to Users

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "**AS IS**."

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. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, 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 publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received 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 interpretation requests except in those cases where the matter has previously received formal consideration. 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, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to: Secretary, IEEE-SA Standards Board, 445 Hoes Lane, Piscataway, NJ 08854, USA.

Laws and regulations: Users of these documents should consult all applicable laws and regulations. Compliance with the provisions of this standard 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: This document is copyrighted by the IEEE. It is made available 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 this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents: Users of IEEE standards 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. 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 Standards Association Web site at http://ieeexplore.ieee.org/xpl/standards.jsp, or contact the IEEE at the address listed previously.
For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEESA Web site at http://standards.ieee.org.

Errata: Errata, if any, for this and all other standards can be accessed at the following URL:
http://standards.ieee.org/reading/ieee/updates/errata/index.html. Users are encouraged to check this URL for errata periodically.

Interpretations: Current interpretations can be accessed at the following URL: http://standards.ieee.org/reading/ieee/interp/index.html.

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 with respect to the existence or validity of any patent rights in connection therewith. 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/24765/24765-2010/24765-2010_wg-participants.pdf.

IMPORTANT NOTICE: This standard is not intended to ensure safety, security, health, or environmental protection. Implementers of the standard are responsible for determining appropriate safety, security, environmental, and health practices or regulatory requirements.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Documents." They can also be obtained on request from IEEE or viewed at http://standards.ieee.org/IPR/disclaimers.html.

Abstract: ISO/IEC/IEEE 24765:2010 provides a common vocabulary applicable to all systems and software engineering work. It was prepared to collect and standardize terminology. ISO/IEC/IEEE 24765:2010 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. ISO/IEC/IEEE 24765:2010 includes references to the active source standards for each definition so that the use of the term can be further explored.

Keywords: computer, dictionary, information technology, software engineering, systems engineering, terminology, vocabulary.

ICS 01.040.35; 35.080
ISBN 978-0-7381-6205-8 STD96038 (PDF); 978-0-7381-6206-5 STDPD96038 (Print)
Price based on 401 pages