INTERNATIONAL STANDARD

ISO/IEC 20944-1

First edition
2013-01-15

Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) —

Part 1: Framework, common vocabulary, and common provisions for conformance

Technologies de l'information — Interopérabilité et liaisons des registres de métadonnées (MDR-IB) —

Partie 1: Cadre d'applications, vocabulaire commun et dispositions communes de conformité

Reference number
ISO/IEC 20944-1:2013(E)

Contents

Page

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.

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

The main task of the joint technical committee 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 drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 20944-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange*.

ISO/IEC 20944 consists of the following parts, under the general title *Information technology — Metadata Registries Interoperability and Bindings (MDR-IB)*:

— *Part 1: Framework, common vocabulary, and common provisions for conformance*

— *Part 2: Coding bindings*

— *Part 3: API bindings*

— *Part 4: Protocol bindings*

— *Part 5: Profiles*

Introduction

ISO/IEC 20944 provides the bindings and their interoperability for metadata registries, such as those specified in ISO/IEC 11179.

This part of ISO/IEC 20944 contains an overview, framework, common vocabulary, and common provisions for conformance for ISO/IEC 20944. In the context of increasing metadata and data interoperability harmonization, four methodologies have been employed to simplify the tasks and to reduce risk.

The first methodology employed is the treating of data (and metadata) interoperability as a series of layered technical specifications (e.g., standards), from application-independent layers to application-specific layer(s).

The second methodology employed is the simplification of interoperability specializations, also known as bindings. Rather than independently developing each separate method of representation and access [codings, application programming interfaces (APIs), protocols], a common, harmonized approach is taken where each binding is derived in a consistent two-step process:

— Step #1 is choosing from the categories of coding, API, protocol (or combination), which themselves are derived from a common data model and navigation method.

— Step #2 is to derive the specific binding from its general binding, e.g., the XML coding binding (ISO/IEC 20944-2:2012, Clause 12) and other (specific) coding bindings are derived from the generic coding binding (ISO/IEC 20944-2:2012, Clauses 1-10); the C API binding (ISO/IEC 20944-3:2012, Clause 11), the Java API binding (ISO/IEC 20944-3:2012, Clause 12), and the other API bindings are derived from the generic API binding (ISO/IEC 20944-3:2012, Clauses 1-10). Because these bindings have a well-defined derivation, the bindings are harmonized, i.e., there is commonality in meaning and interpretation across the bindings. Thus, the complexity of adding and harmonizing a new (coding, API, protocol) binding is greatly simplified.

The third methodology employed is the use of rule-based bindings to simplify the normative wording of the standards. A rule-based binding is a binding that is specified by a general set of rules (in contrast to application-specific normative wording). For example, the XML coding binding is based upon a set of transformation rules (in contrast to specifying a specific DTD or XML schema).

The fourth methodology involves the harmonization of bindings within a category. For example, the XML coding binding is intended to be harmonized with the ASN.1 coding binding; the C API binding is intended to be harmonized with the Java API binding, etc.

Information technology — Metadata Registries Interoperability and Bindings (MDR-IB) —

Part 1: Framework, common vocabulary, and common provisions for conformance

1 Scope

1.1 General

ISO/IEC 20944 is a series of International Standards that describe codings, APIs, and protocols for interacting with an ISO/IEC 11179 metadata registry (MDR).

This part of ISO/IEC 20944 provides the overview, framework, common vocabulary, and common provisions for conformance for ISO/IEC 20944. It addresses the following data interoperability features[1]:

— a common framework for variety control: harmonized concepts for conforming implementations and strictly conforming implementations;

— harmonized provisions, such as mandatory requirements[2] and optional requirements[3], and their consistent application across all bindings of ISO/IEC 20944;

— harmonized and consistent treatment of data elements with varying data obligation attributes (e.g., mandatory, conditional, optional, extended) and varying data longevity attributes (e.g., in-use, obsolete, reserved, etc.).

This part of ISO/IEC 20944 also includes a rationale that guided its development. The rationale also discusses the harmonized use of profiles (e.g., subsets, supersets, changes, etc.) of the data structure and data elements.

[1] The concept of *data interoperability* applies to metadata when metadata is treated as data, e.g., metadata item attributes (as specified by ISO/IEC 11179-3) that are transferred or exchanged. The concept of data interoperability is different from *metadata interoperability* (agreement upon the meaning of descriptive data), which is outside the scope of ISO/IEC 20944.

[2] In the context of this part of ISO/IEC 20944, the term *mandatory requirement* is defined as in ISO/IEC Guide 2:2004, subclause 7.5.1: *a requirement of a normative document that must necessarily be fulfilled in order to comply with that document*. There is *no implication* that the aforementioned requirement is compulsory by law or regulation. This kind of *mandatory requirement* is also known as an *exclusive requirement*.

[3] ISO/IEC Guide 2:2004, subclause 7.5.2 defines the term *optional requirement*, which includes the following 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 that may otherwise be disregarded.*

1.2 Overview of concepts

1.2.1 Metadata vs. data

Metadata is descriptive data about objects[4]. The *essential characteristics* of metadata include: it is *descriptive data*, and that it is *descriptive about something*. For example, if ***P*** is data and ***P→Q*** represents the descriptive relationship such that ***P*** describes ***Q***, then ***P*** is metadata about ***Q***. If there is no relationship from ***P*** to ***Q***, then ***P*** is no longer metadata (i.e., ***P*** is merely data) because metadata is always relative to the object of description. Or stated differently, ***P*** only becomes metadata once its descriptive relationship to ***Q*** is established. Thus, it is *impossible to determine, independent of context and relationships*, that any piece of data is actually also metadata. The implications are: (1) because metadata is data, it can be exchanged like other data, but (2) to remain metadata, the exchange must include the associated context and relationships. ISO/IEC 20944 simply treats everything as data — whether it is *used as metadata* is outside the scope of ISO/IEC 20944. Although metadata is just data, ISO/IEC 20944 also provides reification[5] and navigation of these contexts and relationships that are particular to metadata (and atypical for common data sets).

NOTE ISO/IEC 20944-5 provides a mapping and a profile such that ISO/IEC 20944 bindings may be used to interchange metadata contained in ISO/IEC 11179 metadata registries, e.g., an application may connect to, access, read, and use metadata from an ISO/IEC 11179 metadata registry.

1.2.2 Metadata and data interoperability

The successful interchange of data is dependent upon mutual agreement of interchange participants. Some key requirements for successful data interchange include (from lower implementation details to higher level abstractions):

— The *syntax* determines how data is coded (structured) and encoded (represented). Codings include specifications for organizing data structures (e.g., *How are records represented? Is tagging embedded or implied?*). Encodings include specifications for representation of datatypes (e.g., are numbers represented as a string of characters or a string of bits?).

EXAMPLE 1 In XML, *"the temperature is 17°"* might be *coded* as a tagged element "<temp>17</temp>" that is *encoded* as 15 UTF-8 characters, the encoding would be the ordering of the bits within the octet, e.g., little endian vs. big endian.

EXAMPLE 2 In the programming language C, *"the temperature is 17°"* might be *coded* as a single binary octet { uint8_t temp = 17; }, and *encoded* as a two's complement big-endian 16-bit integer.

— The *semantics* define the meaning of the data. Several kinds of descriptive techniques are possible, such as using ISO/IEC 11179-3 for describing data. Additional technical specifications, such as standards, may be used in conjunction with the ISO/IEC 11179-3 description of data.

EXAMPLE 3 The statement *"the temperature is 17°"* might not be descriptive enough because (1) it does not convey units of measure, e.g., Celsius or Fahrenheit; and (2) it does not convey what is being measured, e.g., temperature sensor #289. Both these features are part of the semantic description that comprises an ISO/IEC 11179-3 Data Element.

— Application-specific behavior is determined by the context of the data.

EXAMPLE 4 The statement *"temperature is 17°C at sensor #289"* may have different meanings depending upon the application. In a telemetry application, the statement *"temperature is 17°C at sensor #289"* might represent data to be recorded and analyzed, such as updating low, average, and high values in a set of time-series data. In contrast, in a heating, ventilation, and air conditioning (HVAC) application, the statement *"temperature is 17°C at sensor #289"* might represent a signal that causes heating units to turn on automatically.

4 For example, metadata may be descriptive data about other data.

5 Reification is to transform into data, e.g., a relationship between datums is transformed into data itself.

Of the three issues above, ISO/IEC 20944 concerns itself with the syntax, i.e., the bindings (codings, APIs, and protocols) for data interchange.

Regarding the semantics, the ISO/IEC 11179 series is a primary tool for specifying semantics, via descriptive data, for data interchange.[6] This descriptive data is known as metadata. The descriptive data (metadata) may also be interchanged via the ISO/IEC 20944 series. However, in this case the ISO/IEC 20944 series is being used for a different purpose: descriptive data interchange (i.e., metadata interchange) rather than data interchange. It is possible to have separate data and metadata interchanges, and to use the ISO/IEC 20944 series independently for each interchange.

Neither ISO/IEC 20944 nor ISO/IEC 11179 specify application-specific requirements and functionality.

1.2.3 Achieving metadata and data interoperability and harmonization

Interoperability with a metadata registry can be achieved in various ways. ISO/IEC 20944 provides a framework within which several approaches can be standardized. All interoperability requires some kind of interface, and associated bindings, between two or more participating functional units. A binding provides a concrete mapping of a functional unit to an interface. Three categories of bindings are supported by ISO/IEC 20944:

— codings, which deal with the formalized representation of information;

— APIs, which specify a binding in programming terms;

— protocols, which specify formalized communications.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC Guide 2, *Standardization and related activities — General vocabulary*

ISO/IEC JTC1 Directives, 5th edition

ISO/IEC 704:2000, *Terminology work — Principles and methods*

ISO/IEC 1087-1:2000, *Terminology work — Vocabulary — Part 1: Theory and application*

ISO/IEC 2382 (all parts), *Information technology — Vocabulary*

ISO/IEC TR 10000-1, *Information technology — Framework and taxonomy of International Standardized Profiles — Part 1: General principles and documentation framework*

ISO/IEC 10241:1992, *International terminology standards — Preparation and layout*

ISO/IEC 11179 (all parts), *Information technology — Metadata registries (MDR)*

ISO/IEC 11404:2007, *Information technology — General-Purpose Datatypes (GPD)*

ISO/IEC 13886:1996, *Information technology — Language-Independent Procedure Calling (LIPC)*

ISO/IEC 14977:1996, *Information technology — Syntactic metalanguage — Extended BNF*

ISO/IEC 19501:2005, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*

[6] Additional semantics may be described by or supplanted with unstructured descriptive text and/or layering of additional standards and specifications.

3 Terms and definitions

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

3.1 Terms and definitions from ISO/IEC Guide 2

Definitions are taken verbatim from ISO/IEC Guide 2 unless otherwise stated in a following note.

3.1.1 Standardization

3.1.1.1
standardization
activity of establishing, with regard to actual or potential problems, provisions for common and repeated use, aimed at the achievement of the optimum degree of order in a given context

NOTE 1 In particular, the activity consists of the processes of formulating, issuing and implementing standards.

NOTE 2 Important benefits of standardization are improvement of the suitability of products, processes and services for their intended purposes, prevention of barriers to trade and facilitation of technological cooperation.

3.1.1.2
level of standardization
geographical, political or economic extent of involvement in standardization

3.1.1.3
consensus
general agreement, characterized by the absence of sustained opposition to substantial issues by any important part of the concerned interests and by a process that involves seeking to take into account the views of all parties concerned and to reconcile any conflicting arguments

3.1.2 Aims of standardization

3.1.2.1
fitness for purpose
ability of a product, process or service to serve a defined purpose under specific conditions

3.1.2.2
compatibility
suitability of products, processes or services for use together under specific conditions to fulfil relevant requirements without causing unacceptable interactions

3.1.2.3
interchangeability
ability of one product, process or service to be used in place of another to fulfil the same requirements

NOTE The functional aspect of interchangeability is called "functional interchangeability", and the dimensional aspect "dimensional interchangeability".

3.1.2.4
variety control
selection of the optimum number of sizes or types of products, processes or services to meet prevailing needs

NOTE Variety control is usually concerned with variety reduction.

3.1.3 Normative documents

3.1.3.1
normative document
document that provides rules, guidelines or characteristics for activities or their results

NOTE 1 The term "normative document" is a generic term that covers such documents as standards, technical specifications, codes of practice and regulations.

NOTE 2 A "document" is to be understood as any medium with information recorded on or in it.

NOTE 3 The terms for different kinds of normative documents are defined considering the document and its content as a single entity.

3.1.3.2
standard
document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context

NOTE Standards should be based on the consolidated results of science, technology and experience, and aimed at the promotion of optimum community benefits.

3.1.3.3
international standard
standard that is adopted by an international standardizing/standards organization and made available to the public

3.1.3.4
regional standard
standard that is adopted by a regional standardizing/standards organization and made available to the public

3.1.3.5
national standard
standard that is adopted by a national standards body and made available to the public

3.1.3.6
provincial standard
standard that is adopted at the level of a territorial division of a country and made available to the public

3.1.3.7
prestandard
document that is adopted provisionally by a standardizing body and made available to the public in order that the necessary experience may be gained from its application on which to base a standard

3.1.3.8
technical specification
document that prescribes technical requirements to be fulfilled by a product, process or service

NOTE 1 A technical specification should indicate, whenever appropriate, the procedure(s) by means of which it may be determined whether the requirements given are fulfilled.

NOTE 2 A technical specification may be a standard, a part of a standard or independent of a standard.

3.1.3.9
regulation
document providing binding legislative rules, that is adopted by an authority

3.1.3.10
technical regulation
regulation that provides technical requirements, either directly or by referring to or incorporating the content of a standard, technical specification or code of practice

NOTE A technical regulation may be supplemented by technical guidance that outlines some means of compliance with the requirements of the regulation, i.e. deemed-to-satisfy provision.

3.1.4 Harmonization of standards

3.1.4.1
harmonized standards
equivalent standards
standards on the same subject approved by different standardizing bodies, that establish interchangeability of products, processes and services, or mutual understanding of test results or information provided according to these standards

NOTE Within this definition, harmonized standards might have differences in presentation and even in substance, e.g. in explanatory notes, guidance on how to fulfil the requirements of the standard, preferences for alternatives and varieties.

3.1.4.2
unified standards
harmonized standards that are identical in substance but not in presentation

3.1.4.3
identical standards
harmonized standards that are identical in both substance and presentation

NOTE 1 Identification of the standards may be different.

NOTE 2 If in different languages, the standards are accurate translations.

3.1.4.4
comparable standards
standards on the same products, processes or services, approved by different standardizing bodies, in which different requirements are based on the same characteristics and assessed by the same methods, thus permitting unambiguous comparison of differences in the requirements

NOTE Comparable standards are not harmonized (or equivalent) standards.

3.1.5 Content of normative documents

3.1.5.1
provision
expression of normative wording that takes the form of a statement, an instruction, a recommendation or a requirement

NOTE These types of provision are distinguished 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".

3.1.5.2
statement
provision that conveys information

3.1.5.3
instruction
provision that conveys an action to be performed

3.1.5.4
recommendation
provision that conveys advice or guidance

3.1.5.5
requirement
provision that conveys criteria to be fulfilled

3.1.5.6
exclusive requirement
mandatory requirement (deprecated)
requirement of a normative document that must necessarily be fulfilled in order to comply with that document

NOTE The term "mandatory requirement" should be used to mean only a requirement made compulsory by law or regulation.

3.1.5.7
optional requirement
requirement of a normative document that must be fulfilled in order to comply with a particular option permitted by that document

NOTE An optional requirement may be either: (1) one of two or more alternative requirements; or (2) an additional requirement that must be fulfilled only if applicable and that may otherwise be disregarded.

3.1.5.8
deemed-to-satisfy provision
provision that indicates one or more means of compliance with a requirement of a normative document

3.1.5.9
descriptive provision
provision for fitness for purpose that concerns the characteristics of a product, process or service

NOTE A descriptive provision usually conveys design, constructional details, etc. with dimensions and material composition.

3.1.5.10
performance provision
provision for fitness for purpose that concerns the behavior of a product, process or service in or related to use

3.1.6 Implementation of normative documents

3.1.6.1
adoption of an international standard (in a national normative document)
publication of a national normative document based on a relevant international standard, or endorsement of the international standard as having the same status as a national normative document, with any deviations from the international standard identified

NOTE ISO/IEC Guide 2 uses the term "taking over of an international standard (in a national normative document)". ISO/IEC Guide 2 explains that "taking over of ..." is equivalent to "adoption of ...".

3.1.6.2
application of a normative document
use of a normative document in production, trade, etc.

3.1.6.3
direct application of a normative document
application of an international standard irrespective of the taking over of that international standard in any other normative document

3.1.6.4
indirect application of a normative document
application of an international standard through the medium of another normative document in which it has been taken over

3.1.7 References to standards

3.1.7.1
reference to standards (in regulations)
reference to one or more standards in place of detailed provisions within a regulation

NOTE 1 A reference to standards is either dated, undated or general, and at the same time either exclusive or indicative.

NOTE 2 A reference to standards may be linked to a more general legal provision referring to the state of the art or acknowledged rules of technology. Such a provision may also stand alone.

3.1.7.2
dated reference (to standards)
reference to standards that identifies one or more specific standards in such a way that later revisions of the standard or standards are not to be applied unless the reference is modified

NOTE The standard is usually identified by its number and either date or edition. The title may also be given.

3.1.7.3
undated reference (to standards)
reference to standards that identifies one or more specific standards in such a way that later revisions of the standard or standards are to be applied without the need to modify the reference

NOTE The standard is usually identified only by its number. The title may also be given.

3.1.7.4
general reference (to standards)
reference to standards that designates all standards of a specified body and/or in a particular field without identifying them individually

3.1.7.5
mandatory standard
standard the application of which is made compulsory by virtue of a general law or exclusive reference in a regulation

3.1.8 Conformity in general

3.1.8.1
conformity
fulfillment by a product, process, or service of specified requirements

3.1.8.2
conformity assessment
any activity concerned with determining directly or indirectly that relevant requirements are fulfilled

NOTE Typical examples of conformity assessment activities are sampling, testing and Inspection; evaluation, verification and assurance of conformity (supplier's declaration, certification); registration, accreditation and approval as well as their combinations.

3.2 Terms and definitions from ISO/IEC Directives, Part 2

Definitions are taken verbatim from ISO/IEC Directives, Part 2 unless otherwise stated in a following note.

3.2.1 Fundamental terms

3.2.1.1
International Standard
international standard where the international standards organization is ISO or IEC

3.3 Terms and definitions from ISO/IEC JTC 1 Directives

Definitions are taken verbatim from the ISO/IEC JTC 1 Directives unless otherwise stated in a following note.

3.3.1 Fundamental terms

3.3.1.1
API
application programming interface
boundary across which application software uses facilities of programming languages to invoke services

[JTC 1 Directives, 5th edition, Annex J; and ISO/IEC 13886]

3.4 Terms and definitions from ISO 704

Definitions are taken verbatim from ISO 704 unless otherwise stated in a following note.

3.4.1 Basic concepts

3.4.1.1
object
<terminology> anything that may be perceived or conceived

3.4.1.2
symbol
designation that is non-linguistic

[adapted[7] from ISO 704]

3.4.1.3
sign
term, appellation, or symbol

[adapted[8] from ISO 704 and ISO 1087-1]

3.5 Terms and definitions from ISO 1087-1

Definitions are taken verbatim from ISO 1087-1 unless otherwise stated in a following note.

3.5.1 Basic concepts

3.5.1.1
definition
representation of a concept by a descriptive statement which serves to differentiate it from related concepts

3.5.1.2
designation
representation of a concept by a sign which denotes it

EXAMPLES Budget Amount (a term, according to ISO 704), New York City (an appellation, according to ISO 704), AHRS1 (a symbol, according to ISO 704).

NOTE A designation may be human readable (e.g., Budget Amount, grand_total) or not (e.g., AHRS1, 1.3.60.1234).

---

[7] The definition is adapted from the text in ISO 704:2000, subclause 7.5.

[8] The definition of *sign* is adapted from ISO 704:2000, subclause 7.5 on the definition of *symbol*, and from the definitions of *designation*, *term*, and *appellation* in ISO 1087-1.

3.6 Terms and definitions from ISO/IEC 2382-1, fundamental terms

Definitions are taken verbatim from ISO/IEC 2382-1 unless otherwise stated in a following note.

3.6.1 General terms

3.6.1.1
text
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

EXAMPLES A business letter printed on paper or displayed on a screen.

3.6.1.2
to access
to obtain the use of a resource

3.6.1.3
information processing
systematic performance of operations upon information, that includes data processing and may include operations such as data communication and office automation

NOTE The term information processing is not a synonym for data processing. Information processing includes data communication (e.g., computer networks) and office automation (e.g., satisfying the business needs of an entity), whereas data processing does not include data communication and office automation.

3.6.1.4
data processing
DP
automatic data processing
ADP
systematic performance of operations upon data

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 is not a synonym for information processing. Information processing includes data communication (e.g., computer networks) and office automation (e.g., satisfying the business needs of an entity), whereas data processing does not include data communication and office automation.

3.6.1.5
software
all or part of the programs, procedures, rules, and associated documentation of an information processing system

NOTE Software is an intellectual creation that is independent of the medium on which it is recorded.

3.6.1.6
storage
<device> functional unit into which data can be placed, in which they can be retained, and from which they can be retrieved

3.6.1.7
data processing system
computer system
computing system
one or more computers, peripheral equipment, and software that perform data processing

3.6.1.8
information processing system
one or more data processing systems and devices, such as office and communication equipment, that perform information processing

3.6.1.9
information system
information processing system, together with associated organizational resources such as human, technical, and financial resources, that provides and distributes information

3.6.1.10
process (1)
predetermined course of events defined by its purpose or by its effect, achieved under given conditions

3.6.1.11
process (2)
<data processing> predetermined course of events that occur during the execution of all or part of a program

3.6.1.12
configuration
manner in which the resources of an information processing system are organized and interconnected

3.6.1.13
synchronous
pertaining to two or more processes that depend upon the occurrence of specific events such as common timing signals

3.6.1.14
asynchronous
pertaining to two or more processes that do not depend upon the occurrence of specific events such as common timing signals

3.6.1.15
input (1), noun
<data> data entered into an information processing system or any of its parts for storage or processing

3.6.1.16
input (2), verb
<process> process of entering data into an information processing system or any of its parts for storage or processing

3.6.1.17
input (3), adj.
pertaining to a device, process, or input-output channel involved in an input process, or to the associated data or states

NOTE The word "input" may be used in place of "input data", "input signal", or "input process" when such a usage is clear in a given context.

3.6.1.18
output (1), noun
<data> data that an information processing system, or any of its parts, transfers outside of that system or part

3.6.1.19
output (2), verb
<process> process by which an information processing system, or any of its parts, transfers data outside of that system or part

3.6.1.20
output (3), adj.
pertaining to a device, process, or input-output channel involved in an output process, or to the associated data or states

NOTE The word "output" may be used in place of "output data", "output signal", or "output process" when such a usage is clear in a given context.

3.6.1.21
interface
shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical interconnections, signal exchanges, and other characteristics, as appropriate

3.6.1.22
data communication
transfer of data among functional units according to sets of rules governing data transmission and the coordination of the exchange

3.6.1.23
functional unit
entity capable of accomplishing a specified purpose

EXAMPLES A hardware subsystem, a software component, both, a human operator, all three (hardware, software, operator).

3.6.1.24
interoperability
capability to communicate, execute programs, or 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

3.6.1.25
virtual
pertaining to a functional unit that appears to be real, but whose functions are accomplished by other means

3.6.1.26
virtual machine
VM
virtual data processing system that appears to be at the exclusive disposal of a particular user, but whose functions are accomplished by sharing the resources of a real data processing system

3.6.1.27
data medium
material in or on which data can be recorded and from which data can be retrieved

3.6.1.28
to log on
to log in
to initiate a session

3.6.1.29
to log off
to log out
to end a session

3.6.2 Information representation

3.6.2.1
numeric
numerical
pertaining to data that consist of numerals as well as to processes and functional units that use those data

3.6.2.2
alphanumeric
pertaining to data that consist of letters, digits, and usually other characters, such as punctuation marks, as well as to processes and functional units that use those data

3.6.2.3
bit
binary digit
either of the digits 0 or 1 when used in the binary numeration system

3.6.2.4
byte
string that consists of a number of bits, treated as a unit, and usually representing a character or a part of a character

NOTE 1 The number of bits in a byte is fixed for a given data processing system.

NOTE 2 The number of bits in a byte is usually 8.

3.6.2.5
octet
8-bit byte
byte that consists of eight bits

3.6.3 Software

3.6.3.1
application software
application program
software or a program that is specific to the solution of an application problem

EXAMPLE A spreadsheet program

3.6.3.2
system software
application-independent software that supports the running of application software

EXAMPLE An operating system

3.6.3.3
support software
support program
software or a program that aids in the development, maintenance, or use of other software or provides general application-independent capability

EXAMPLE A compiler, a database management system

3.6.3.4
system documentation
collection of documents that describe the requirements, capabilities, limitations, design, operation, and maintenance of an information processing system

3.6.3.5
application documentation
collection of documents that describe the requirements, capabilities, limitations, design, operation, and maintenance of application software or an application program

EXAMPLE User and installation documentation for a program

3.6.3.6
portability (of a program or of data)
capability to be interpreted, understood, or executed on various types of data processing systems without conversion and with little or no modification

3.6.3.7
operating system
OS
software that controls the execution of programs and that may provide services such as resource allocation, scheduling, input-output control, and data management

3.6.4 Programming

3.6.4.1
program
computer program
syntactic unit that conforms to the rules of a particular programming language and that is composed of declarations and statements or instructions needed to solve a certain function, task, or problem

3.6.4.2
programming
designing, writing, modifying, and testing of programs

3.6.4.3
artificial language
language whose rules are explicitly established prior to its use

3.6.4.4
programming language
artificial language for expressing programs

3.6.5 Computer security

3.6.5.1
data protection
implementation of appropriate administrative, technical or physical means to guard against unauthorized intentional or accidental disclosure, modification, or destruction of data

3.6.6 Data management

3.6.6.1
information management
functions of controlling the acquisition, analysis, retention, retrieval, and distribution of information all within an information processing system

3.6.6.2
data management
functions that provide access to data, performs or monitors the storage of data, and controls input-output operations all within a data processing system

3.6.6.3
access method
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 method to transfer data

EXAMPLE 1 Random access method, indexed access method, sequential access method

EXAMPLE 2 In object-oriented design, any class method that sends or receives data

3.6.6.4
database
collection of data organized according to a conceptual structure describing the characteristics of these data and the relationships among their corresponding entities, supporting one or more application areas

3.6.6.5
file
labeled set of records treated as a unit

3.7 Terms and definitions from ISO/IEC 2382-4, organization of data

Definitions are taken verbatim from ISO/IEC 2382-4 unless otherwise stated in a following note.

3.7.1 Character sets

3.7.1.1
character
member of a set of elements that is used for the representation, organization, or control of data

NOTE Characters may be categorized as follows:

- character
  - graphic character
    - digit
    - letter
    - ideogram
    - special character
  - control character
    - transmission control character
    - format effector
    - code extension character
    - device control character

EXAMPLE The international reference version of the character set of ISO/IEC 10646

3.7.1.2
character set
finite set of characters that is complete for a given purpose

EXAMPLE The international reference version of the character set of ISO/IEC 10646

3.7.1.3
alphabetic character set
character set that contains letters and may contain special characters, but not digits

EXAMPLE The international reference version of the character set of ISO/IEC 10646

3.7.1.4
numeric character set
character set that contains digits and may contain special characters, but usually not letters

3.7.1.5
alphanumeric character set
character set that contains both letters and digits and may contain special characters

3.7.1.6
binary character set
character set that consists of two characters

3.7.2 Codes

3.7.2.1
code, noun
coding scheme
collection of rules that maps the elements of a first set onto the elements of a second set

NOTE 1 The elements of either set may be characters or character strings.

NOTE 2 The first set is called coded set and the second set is called code set.

NOTE 3 Each element of the code set may be related to more than one element of the coded set but the reverse is not true.

3.7.2.2
coded set
set of elements that is mapped onto another set according to a code

EXAMPLE A list of the names of airports that is mapped onto a corresponding set of three-letter abbreviations

3.7.2.3
code value
code element
code (deprecated)
result of applying a code to an element of a coded set

EXAMPLES “CDG” representing Paris Charles-De-Gaulle in the code for three-letter representation of airport names; the hexadecimal number 0041 representing “Latin capital letter A” in ISO/IEC 10646

3.7.2.4
code set
code element set
code (deprecated)
result of applying a code to all elements of a coded set

EXAMPLES All the three-letter representations of airport names

3.7.2.5
alphabetic code
code whose application results in an alphabetic code set

3.7.2.6
numeric code
code whose application results in a numeric code set

3.7.2.7
alphanumeric code
code whose application results in an alphanumeric code set

3.7.2.8
binary code
code whose application results in a binary code set

3.7.2.9
alphabetic code set
alphabetic code element set
code set whose elements are taken from an alphabetic character set

3.7.2.10
numeric code set
numeric code element set
code set whose elements are constructed from a numeric character set

3.7.2.11
alphanumeric code set
alphanumeric code element set
code set whose elements are constructed from an alphanumeric character set

3.7.2.12
binary code set
binary code element set
code set whose elements are constructed from a binary character set

3.7.3 Graphic characters

3.7.3.1
graphic character
character, other than a control character, that has a visual representation and is normally produced by writing, printing, or displaying on a screen

NOTE A graphic character may be used to represent a control character in text.

3.7.3.2
letter
alphabetic character
graphic character that, when appearing alone or combined with others, represents one or more concepts of a written language, or one or more sound elements of a spoken language

NOTE Diacritical marks used alone and punctuation marks are not considered to be letters.

3.7.3.3
alphabet
character set in which the order of its elements has been agreed upon

EXAMPLE The 128 ASCII characters

3.7.3.4
ideogram
ideographic character
graphic character, in a natural language, that represents an object or a concept and associated sound elements

EXAMPLES A Chinese ideogram or a Japanese Kanji

3.7.3.5
digit
numeric character
character that represents a natural number

EXAMPLES One of the characters 0 through 9 in the decimal system; these digits plus the characters A through F used in the hexadecimal system

NOTE 1 The mathematical term “natural number” denotes all non-negative integers.

NOTE 2 This is a modified version of the definition in ISO/IEC 2382-1.

3.7.3.6
alphanumeric character
character of an alphanumeric character set

3.7.3.7
decimal digit
digit used in the decimal system

EXAMPLE The Arabic digits 0 through 9

3.7.3.8
hexadecimal digit
digit used in the hexadecimal system

EXAMPLE The graphic characters 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

3.7.3.9
bit
binary digit
either of the digits 0 or 1 when used in the binary system

3.7.3.10
binary character
either character of a binary character set

EXAMPLES T (true) or F (false), Y (yes) or N (no)

3.7.3.11
blank
blank character
character that represents an empty position in a graphic character string

NOTE 1 A blank is conceptually different from a space character but may not be differentiated in a particular character set. For example, some character sets include a blank as a "nonbreaking space" that may be used between two graphic characters without being treated as a delimiter.

NOTE 2 Blank is not included in ISO/IEC 10646.

3.7.3.12
special character
graphic character that is neither a letter, digit, nor blank, and usually not an ideogram

EXAMPLES A punctuation mark, a percent sign, a mathematical symbol

3.7.4 Control characters

3.7.4.1
control character
character whose purpose is to effect format, to control data transmission, or to perform other control functions

NOTE A control character, although it is not a graphic character, may have a graphic representation.

3.7.4.2
transmission control character
control character used to control or facilitate data transmission between data terminal equipment

NOTE Transmission control characters are described in ISO/IEC 10646 and ISO 6429.

3.7.4.3
space character
character that causes the print or display position to advance one position along the line without producing any graphic character

NOTE The space character is described in ISO/IEC 10646, ISO 4873 and ISO 6937-1.

3.7.4.4
format effector
control character used to position printed, displayed, or recorded data

NOTE Format effectors are described in ISO/IEC 10646 and ISO 6429.

EXAMPLES Space character, horizontal-tab character, carriage-return character, line-feed character, and form-feed character

3.7.4.5
code extension character
control character used to indicate that one or more of the succeeding code values are to be interpreted according to a different code

NOTE Code extension characters are described in ISO/IEC 10646 and ISO 2022.

3.7.4.6
device control character
control character used to specify a control function for peripheral equipment associated with a data processing system

NOTE Device control characters are described in ISO/IEC 10646 and ISO 6429.

3.7.5 Strings

3.7.5.1
string
sequence of elements of the same nature, such as characters or bits, considered as a whole

NOTE A string may be empty or contain only one element.

3.7.5.2
character string
string consisting solely of characters

3.7.5.3
alphabetic string
string consisting solely of letters from the same alphabet

3.7.5.4
bit string
string consisting solely of bits

3.7.5.5
null string
string that contains no elements

3.7.5.6
position
location of an element in a string

3.7.5.7
n-bit byte
byte with an explicit specification of the number of its bits

EXAMPLE 7-bit byte

3.7.6 Words

3.7.6.1
word
character string or a bit string treated as a unit for a given purpose

NOTE The length of a computer word is defined by the computer architecture, while the words in text processing are delimited by special characters or control characters.

3.7.6.2
alphabetic word
word that consists of characters from the same alphabetic character set

3.7.6.3
numeric word
word that consists of digits and possibly space characters and special characters

EXAMPLE The string "33 (1) 41 62 80 00" as written for the AFNOR telephone number

3.7.6.4
alphanumeric word
word that consists of characters from the same alphanumeric character set

3.7.6.5
computer word
machine word
word, usually treated as a unit, that is suitable for processing by a given computer

3.7.6.6
word length
word size
number of characters or bits in a word

3.7.7 Structuring of data

3.7.7.1
data element
<organization of data> unit of data that is considered, in context, to be indivisible

EXAMPLE 1 The data element “age of a person” with values consisting of all combinations of 3 decimal digits

EXAMPLE 2 A personnel record that includes the data elements "name" and "address". In the context of the personnel record, "name" and "address" function as an indivisible unit, e.g., the data element "name" and the data element "address" each can be stored and retrieved as an indivisible unit. However, in a *different* context, "address" itself may be considered a record that contains its own data elements "street address", "city", "postal code", "country"

3.7.7.2
record
<organization of data> set of data elements treated as a unit

3.7.7.3
table
arrangement of data in which each item may be referenced by means of arguments or keys

3.7.8 Delimiters and identifiers

3.7.8.1
internal label
label that is recorded on a data medium and that provides information about data recorded on the data medium

3.7.9 Trees

3.7.9.1
node
<organization of data> point from which subordinate items originate within a data structure

NOTE A node may have no subordinate items and is then called a terminal node.

3.7.9.2
tree
rooted tree
data structure containing nodes that are linked together hierarchically with at most one parent node for each node, and with only one root node

3.7.9.3
subtree
node within a tree with all its connected descendent nodes

3.7.9.4
ordered tree
tree in which the order of the subtrees of each node is significant

3.8 Terms and definitions from ISO/IEC 2382-5, representation of data

Definitions are taken verbatim from ISO/IEC 2382-5 unless otherwise stated in a following note.

3.8.1 Types of data representation

3.8.1.1
predefined
built-in
intrinsic
pertaining to a language construct that is declared by the definition of the artificial language

EXAMPLE The construct **sizeof()** is built-in to the C programming language

3.9 Terms and definitions from ISO/IEC 2382-6, preparation and handling of data

Definitions are taken verbatim from ISO/IEC 2382-6 unless otherwise stated in a following note.

3.9.1 General terms

3.9.1.1
to read
to obtain data from a storage device, from a data medium, or from another source

3.9.1.2
to write
to make a permanent or transient recording of data in a storage device or on a data medium

NOTE The phrases "to read to" and "to read from" are often distinguished from the phrases "to write to" and "to write from" only by the viewpoint of the description. For example, the transfer of a block of data from internal storage to external storage may be called "writing to the external storage" or "reading from the internal storage", or both.

3.9.1.3
to copy
to read data from a source data medium, leaving the source data unchanged, and to write the same data on a destination data medium that may differ from that of the source

EXAMPLE To copy a file from a magnetic tape onto a magnetic disk

3.9.1.4
to duplicate
to copy from a source data medium to a destination data medium that has the same physical form

EXAMPLE To copy a file from a magnetic tape to another magnetic tape

3.9.2 Transfer and conversion

3.9.2.1
to transfer
to move
to send data from one storage location to another

3.9.2.2
to transform
to change the form of data according to specified rules, without fundamentally changing the meaning of the data

3.9.2.3
to translate
to change the form of data from one artificial language into artificial language or into some other representation suitable for processing

3.9.2.4
to convert
to change the representation of data from one form to another, without changing the information conveyed

EXAMPLE Code conversion; radix conversion; analog to digital conversion, media conversion

3.9.2.5
to transliterate
to convert data character by character

3.9.2.6
to encode
to code
to convert data by the use of a code in such a manner that reconversion to the original form is possible

3.9.2.7
to decode
to convert data by reversing the effect of some previous encoding

3.9.2.8
to transcribe
to copy data from one data medium to another, converting them as necessary for acceptance by the receiving medium

3.9.2.9
to pack
to convert data to a compact form in a storage medium by taking advantage of known characteristics of the data and of the storage medium, in such a way that the original form of the data can be recovered

EXAMPLE To make use of bit or byte locations that would otherwise remain unused

3.9.2.10
packing
operation performed when data are packed

3.9.2.11
to unpack
to recover the original form of the data from packed data

3.9.2.12
to compress
to compact
to reduce the space taken on a data medium by encoding or removing repetitive characters

3.9.2.13
to expand
to return compressed data to their original form

3.9.3 Searches

3.9.3.1
search
examination of one or more data elements of a set to find those elements that have a given property

3.9.3.2
search key
key used for data retrieval

3.9.3.3
scanning
systematic examination of data

3.9.4 Ordering, sorting, and collating

3.9.4.1
to order
to place items in an arrangement in accordance with specified rules

3.9.4.2
order
specified arrangement resulting from ordering

NOTE In contrast to a sequence, an order need not be linear, for example the ordering of a hierarchy of items.

3.9.4.3
to sequence
to place items in an arrangement in accordance with the order of the natural numbers

NOTE Methods or procedures may be specified for mapping other natural linear orders onto the natural numbers; then, by extension, sequencing may be, for example, alphabetic or chronological.

3.9.4.4
sequence
series of items that have been sequenced

3.9.4.5
index
list of the contents of a file or of a document, together with keys or references for locating the contents

3.9.5 Preparation of data

3.9.5.1
to edit
to prepare data for a later operation

NOTE Editing may include the rearrangement, the addition or modification of data, the deletion of unwanted data, format control, code conversion, and the application of standard processes such as zero suppression.

3.9.5.2
to extract
to select and remove from a group of items those which meet specific criteria

3.9.5.3
to clear
to cause one or more storage locations to be set in a prescribed state, usually that corresponding to zero or that corresponding to the space character

3.10 Terms and definitions from ISO/IEC 2382-7, computer programming

Definitions are taken verbatim from ISO/IEC 2382-7 unless otherwise stated in a following note.

3.10.1 Kinds of languages

3.10.1.1
expression language
programming language in which assignments can be made in the context of an expression

EXAMPLE C

NOTE The expression "if (x = y < 0) ..." is legal in C, but would be illegal in Ada.

3.10.1.2
markup language
text-formatting language designed to transform raw text into structured documents, by inserting procedural and descriptive markup into the raw text

3.10.1.3
pseudocode
combination of language constructs from a programming language with those of natural language that is not necessarily computer-processible, but intended to make the design of a program manifest to human readers

EXAMPLE

```
IF the data arrive faster than expected,
    THEN reject every third input.
    ELSE process all data received.
ENDIF
```

3.10.2 Program preparation

3.10.2.1
to parse
to determine the syntactic structure of a language construct by decomposing it into lexical tokens and establishing the relationships among them

EXAMPLE To parse blocks into statements, statements into expressions, expressions into operators and operands

3.10.2.2
parser
software tool that parses programs or other text, often as the first step of assembly, compilation, interpretation, or analysis

3.10.3 Program execution

3.10.3.1
language processor
functional unit for translating and executing programs written in a specified programming language

EXAMPLE A LISP machine

3.10.3.2
execution time
run time
any instant at which the execution of a particular program takes place

3.10.3.3
to exit
to execute an instruction or statement in a program or part thereof that terminates the execution of that program or part

3.10.3.4
exit point
point in a program, module, or statement at which execution of this program, module, or statement can terminate

3.10.3.5
entry point
entrance
point in a program, module, or statement at which execution of this program, module, or statement can begin

3.10.3.6
reentry point
point in a program, module, or statement at which this program, module, or statement resumes execution following the execution of another program, module, or statement

3.10.3.7
deadlock
situation in which data processing is suspended because two or more devices or concurrent processes are each awaiting resources assigned to the other(s) or because of other mutual dependencies

EXAMPLE A situation in which a program A, with an exclusive lock on record X, asks for a lock on record Y, which is allocated to program B. Likewise, program B is waiting for exclusive control over record X before giving up control over record Y.

3.10.3.8
lockout
technique for allocation of resources in which shared resources are protected by permitting access by only one device or process at a time and excluding others

EXAMPLE To prohibit reading of data while they are being updated

3.10.3.9
to raise (an exception)
to cause an exception to be signaled based upon the occurrence of a specified condition

3.10.3.10
exception handler
portion of a program executed in response to a specific kind of exception

3.10.3.11
to handle (an exception)
to take direct action as the result of the occurrence of an exception

NOTE Normally, control is transferred to an exception handler that takes action.

3.10.3.12
to propagate (an exception)
to transfer control to the exception handler of a prior calling module or nesting module due to lack of required handling within a given module, or to explicitly raise the exception again within an exception handler

3.10.3.13
data exception
exception that occurs when a program or process attempts to use or access data incorrectly

3.10.3.14
operation exception
exception that occurs when a program or process encounters an invalid operation part

3.10.3.15
protection exception
exception that occurs when a program or process attempts to access a protected area in a storage device or storage medium

3.11 Terms and definitions from ISO/IEC 2382-8, security

Definitions are taken verbatim from ISO/IEC 2382-8 unless otherwise stated in a following note.

3.11.1 General concepts

3.11.1.1
computer security
protection of data and resources from accidental or malicious acts, usually by taking appropriate actions

NOTE These acts may be modification, destruction, access, disclosure, or acquisition, if not authorized.

3.11.1.2
communications security
computer security applied to data communication

3.11.1.3
data security
computer security applied to data

3.11.1.4
security policy
plan or course of action adopted for providing computer security

3.11.1.5
data integrity
property of data whose accuracy and consistency are preserved regardless of changes made

NOTE 1 From an information security perspective, data integrity is a technical policy about information security that reduces inbound security threats to an acceptable level.

NOTE 2 Data integrity may include: controlling the creation of information, controlling changes to information, or other techniques. The policy may be implemented by various security techniques, security technologies, security procedures, practices, etc..

3.11.1.6
file protection
implementation of appropriate administrative, technical, or physical means to guard against the unauthorized access to, modification of, or deletion of a file

3.11.1.7
confidentiality
property of data that indicates the extent to which these data have not been made available or disclosed to unauthorized individuals, processes, or other entities

NOTE 1 Confidentiality is a security technique that minimizes outbound security threats to an acceptable level by permitting retrieval or read access to authorized entities, and prohibiting retrieval and read access to unauthorized entities.

NOTE 2 Various security technologies may implement "confidentiality".

3.11.1.8
authentication
act of verifying the claimed identity of an entity

3.11.1.9
message authentication
verification that a message was sent by the purported originator to the intended recipient and that the message was not changed in transit

3.11.1.10
authentication information
information used to establish the validity of a claimed identity of an entity

3.11.1.11
credentials
data that are transferred to establish the claimed identity of an entity

3.11.1.12
authentication exchange
mechanism intended to ensure the identity of an entity by means of an information exchange

3.11.1.13
authorization
granting of rights, which includes the granting of access based on access rights

3.11.1.14
security level
combination of a hierarchical security classification and a security category that represents the sensitivity of an object or the security clearance of an individual

3.11.1.15
closed-security environment
environment in which special attention is paid (in the form of authorizations, security clearances, configuration controls, etc.) to protect data and resources from accidental or malicious acts

3.11.1.16
open-security environment
environment in which protection of data and resources from accidental or malicious acts is achieved through normal operational procedures

3.11.1.17
subject
<computer systems security> active entity that can access objects

EXAMPLE A process that involves execution of a program

NOTE A subject may cause information to flow among objects or may change the state of the data processing system.

3.11.1.18
object
<computer systems security> entity to which access is controlled

EXAMPLE A file, a program, an area of main storage; data collected and maintained about a person

3.11.2 Classification of information

3.11.2.1
compartmentalization
division of data into isolated blocks with separate security controls for the purpose of reducing risk

EXAMPLE The division of data relative to a major project into blocks corresponding to subprojects, each with its own security protection, in order to limit exposure of the overall project

3.11.3 Cryptographic techniques

3.11.3.1
encryption
encipherment
cryptographic transformation of data

3.11.3.2
irreversible encryption
irreversible encipherment
one-way encryption
encryption that produces ciphertext from which the original data cannot be reproduced

NOTE Irreversible encryption is useful in authentication. For example, a password might be irreversibly encrypted and the resulting ciphertext stored. A password presented later would be irreversibly encrypted identically and the two strings of ciphertext compared. If they are identical, the presented password is correct.

3.11.3.3
decryption
decipherment
process of obtaining, from a ciphertext, the original corresponding data

NOTE A ciphertext may be encrypted a second time, in which case a single decryption does not produce the original plaintext.

3.11.3.4
key
bit string that controls the operations of encryption or decryption

3.11.3.5
private key
key that is intended for decryption for the exclusive use by its owner

3.11.3.6
public key
key that is intended for use by any entity for encrypted communication with the owner of the corresponding private key

3.11.3.7
public-key cryptography
asymmetric cryptography
cryptography in which a public key and a corresponding private key are used for encryption and decryption

NOTE If a public key is used for encryption, the corresponding private key must be used for decryption, and vice versa.

3.11.3.8
symmetric cryptography
cryptography in which the same key is used for encryption and decryption

3.11.4 Access control

3.11.4.1
access control
means of ensuring that the resources of a data processing system can be accessed only by authorized entities in authorized ways

3.11.4.2
access control list
access list
list of entities, together with their access rights, that are authorized to access a resource

3.11.4.3
access right
permission for a subject to access a particular object for a specific type of operation

EXAMPLE Permission for a process to read a file but not write to it

3.11.4.4
access permission
all of a subject's access rights with respect to some object

3.11.4.5
access period
period of time during which specified access rights prevail

3.11.4.6
access type
<computer security> type of operation specified by an access right

EXAMPLES Read, write, execute, append, modify, delete, create

3.11.4.7
ticket
<computer security> representation of one or more access rights that a possessor has to an object

NOTE The ticket represents an access permission.

3.11.4.8
capability
<computer security> representation of the identification of an object, or of a class of objects, and of a set of authorized access types for these objects

NOTE A capability can be implemented in the form of a ticket.

3.11.4.9
capability list
list associated with a subject that identifies all of the subject's access types for all objects

EXAMPLE A list associated with a process that identifies all of its access types for all files and other protected resources

3.11.4.10
identity authentication
identity validation
performance of tests to enable a data processing system to recognize entities

EXAMPLE The checking of a password or of an identity token

3.11.4.11
identity token
device used for identity authentication

EXAMPLE Smart card, metal key

3.11.4.12
password
character string that is used as authentication information

3.11.4.13
logical access control
use of mechanisms related to data or information to provide access control

EXAMPLE The use of a password

3.11.4.14
physical access control
use of physical mechanisms to provide access control

EXAMPLE Keeping a computer in a locked room

3.11.4.15
read access
access right that gives permission to read data

3.11.4.16
write access
access right that gives permission to write data

NOTE Write access may grant permission to append, modify, delete, or create data.

3.11.4.17
user ID
user identification
character string or pattern that is used by a data processing system to identify a user

3.11.4.18
user profile (1)
description of a user, typically used for access control

NOTE A user profile may include data such as user ID, user name, password, access rights, and other attributes.

3.11.4.19
user profile (2)
pattern of a user's activity that can be used to detect changes in the activity

3.11.5 Security violations

3.11.5.1
threat
potential violation of computer security

3.11.5.2
risk
possibility that a particular threat will exploit a particular vulnerability of a data processing system

3.11.5.3
denial of service
prevention of authorized access to resources or the delaying of time-critical operations

3.11.5.4
compromise
violation of computer security whereby programs or data may have been modified, destroyed, or made available to unauthorized entities

3.11.5.5
loss
quantitative measure of harm or deprivation resulting from a compromise

3.11.5.6
exposure
possibility that a particular attack will exploit a particular vulnerability of a data processing system

3.11.5.7
disclosure
violation of computer security whereby data have been made available to unauthorized entities

3.11.5.8
penetration
unauthorized access to a data processing system

3.11.5.9
breach
circumvention or disablement of some element of computer security, with or without detection, which could result in a penetration of the data processing system

3.11.5.10
attack
attempt to violate computer security

EXAMPLES Malicious logic, wiretapping

3.11.5.11
eavesdropping
unauthorized interception of information-bearing emanations

3.11.5.12
aborted connection
disconnection that does not follow established procedures

NOTE An aborted connection may enable other entities to gain unauthorized access.

3.11.5.13
data corruption
accidental or intentional violation of data integrity

3.11.6 Protection of sensitive information

3.11.6.1
verification
comparing an activity, a process, or a product with the corresponding requirements or specifications

EXAMPLES Comparing a specification with a security policy model or comparing object code with source code

3.11.6.2
data validation
process used to determine if data are accurate, complete, or meet specified criteria

NOTE Data validation may include format checks, completeness checks, check key tests, reasonableness checks, and limit checks.

3.11.6.3
digital signature
data appended to a message, that allow the recipient of the message to verify the source of the message

3.11.6.4
digital envelope
data appended to a message, that allow the intended recipient to verify the integrity of the content of the message

3.11.6.5
biometric
pertaining to the use of specific attributes that reflect unique personal characteristics, such as a fingerprint, an eye blood-vessel print, or a voice print, to validate the identity of a person

3.11.6.6
data authentication
process used to verify data integrity

EXAMPLES Verification that data received are identical to data sent, verification that a program is not infected by a virus

NOTE Not to be confused with authentication.

3.11.6.7
message authentication code
bit string that is a function of both data (either plaintext or ciphertext) and a secret key, and that is attached to the data in order to allow data authentication

NOTE The function used to generate the message authentication code is typically a one-way function.

3.11.6.8
repudiation
denial by one of the entities involved in a communication of having participated in all or part of the communication

NOTE In the description of techniques and mechanisms the term “non-repudiation” is often used to mean that none of the entities involved in a communication can deny its participation in the communication.

3.11.6.9
mutual suspicion
relationship between interacting entities in which neither entity relies upon the other entity to function correctly or securely with respect to some property

3.11.6.10
notarization
registration of data with a trusted third party that allows the later assurance of the accuracy of the data’s characteristics such as content, origin, time, and delivery

3.12 Terms and definitions from ISO/IEC 2382-9, data communications

Definitions are taken verbatim from ISO/IEC 2382-9 unless otherwise stated in a following note.

3.12.1 General

3.12.1.1
data transmission
transmission
transfer of data from one point to one or more other points over telecommunication facilities

3.12.1.2
data source
functional unit that originates data for transmission

3.12.1.3
data sink
functional unit that accepts transmitted data

3.12.1.4
transmission medium
natural or artificial medium that conveys signals

3.12.2 Transmission — general

3.12.2.1
to transmit
to send from one location for reception elsewhere

3.12.2.2
simplex transmission
data transmission in one preassigned direction only

3.12.2.3
half-duplex transmission
data transmission in either direction, one direction at a time

3.12.2.4
duplex transmission
full-duplex transmission
data transmission in both directions at the same time

3.12.2.5
asynchronous transmission
data transmission in which the start of each character or block of characters is arbitrary but, once started, the time of occurrence of each signal element has the same relationship to significant instants of a fixed time base

3.12.2.6
synchronous transmission
data transmission in which the time of occurrence of each signal element is related to a fixed time base

3.12.3 Data link

3.12.3.1
data link
parts of two data terminal equipments that are controlled by a protocol along with the interconnecting data circuit, which together enable data transfer

3.12.3.2
data circuit
pair of associated transmission channels that provide a means of two-way data transmission

NOTE 1 Between data switching exchanges, the data circuit may or may not include data circuit-terminating equipment (DCE), depending on the type of interface used at the data switching exchange.

NOTE 2 Between a data station and a data switching exchange or data concentrator, the data circuit includes the data circuit-terminating equipment at the data station end, and may include equipment similar to a DCE at the data switching exchange or data concentrator location.

3.12.3.3
line
transmission line
physical transmission medium

NOTE The line is the portion of a data circuit external to data circuit-terminating equipment (DCE), that connects the DCE to a data switching exchange (DSE), that connects a DCE to one or more other DCEs, or that connects a DSE to another DSE.

3.12.3.4
protocol
set of rules that determines the behavior of functional units in achieving communication

3.12.3.5
error control
part of a protocol that enables error detection, and possibly error correction, of errors

3.12.3.6
flow control
control of the actual transfer rate in data communication

3.12.3.7
acknowledgment
affirmative response, by a receiver, to a sender, indicating that transmitted data have been received

3.12.3.8
polling
process whereby data stations are invited one at a time to transmit

3.12.3.9
data transfer phase
phase of a call during which user data may be transferred between data terminal equipments that are interconnected via a network

NOTE While generally used on a multipoint connection, polling can be used on a point-to-point connection.

3.12.3.10
time-out
event designed to occur at the conclusion of a predetermined elapsed time

NOTE A time-out can be prevented by sending an appropriate signal; a time-out condition can be cancelled by the receipt of an appropriate time-out cancellation signal.

3.12.3.11
recovery
process for resolving conflicting or erroneous conditions arising during the transfer of data in data transmission

3.12.3.12
data station
functional unit that originates data for transmission, that accepts transmitted data, and that performs all functions necessary for communication with another functional unit

3.12.4 Data network

3.12.4.1
node
<data communication> point where one or more functional units interconnect transmission channels or data circuits in a data network

3.12.4.2
port
termination through which signals can enter or leave a network

3.12.4.3
connection
association established between functional units for data transmission

3.12.4.4
point-to-point connection
connection established between two data stations

3.12.4.5
multipoint connection
connection established among more than two data stations

3.12.4.6
broadcast
transmission of the same data to all destinations

3.12.4.7
multicast
transmission of the same data to a selected group of destinations

3.12.4.8
data network
network in which data circuits and possibly switching facilities enable data communication among data terminal equipments

3.12.4.9
wide area network
WAN
network that provides communication services to a geographic area larger than that served by a local area network or a metropolitan area network

3.12.4.10
store and forward
mode of operation of a data network in which data are temporarily stored before they are retransmitted toward the destination

3.12.4.11
packet
sequence of bits arranged in a specific format, containing control data and possibly user data, and that is transmitted and switched as a whole

3.12.4.12
datagram
packet, independent of other packets, that carries information sufficient for routing from the originating data terminal equipment (DTE) to the destination DTE, without relying on earlier exchanges between the DTEs and the network

3.12.4.13
datagram service
service that routes a datagram to the destination identified in its address field without reference by the network to any other datagram

NOTE Datagrams may be delivered to a destination address in a different order from that in which they were entered in the network.

3.12.5 User facilities

3.12.5.1
user class of service
category of a data transmission service provided by a data network in which the data signaling rate, the data terminal equipment operating mode, and the code structure, if any, are standardized

3.12.5.2
user facilities
set of functions available on demand to a user, and provided by a data network as a service for data transmission

NOTE Some user facilities may be available on a per-call basis, and others may be assigned for an agreed period of time at the request of the user.

3.12.5.3
calling
process of transmitting selection signals in order to establish a connection between data stations

3.12.5.4
call
relationship established between data stations that includes establishing a connection, transmitting messages, and terminating the connection

3.12.5.5
answering
process of responding to a calling data station to complete the establishment of a connection between data stations

3.12.5.6
server
functional unit that provides shared services to workstations or to other functional units over a data network

EXAMPLES A file server, a print server, a mail server

3.12.5.7
client
functional unit that receives shared services from a server

3.13 Terms and definitions from ISO/IEC 2382-15, programming languages

Definitions are taken verbatim from ISO/IEC 2382-15 unless otherwise stated in a following note.

3.13.1 Lexical tokens

3.13.1.1
lexical token
lexical element
lexical unit
string of one or more characters of the alphabet of an artificial language that, by convention, represents an elemental unit of meaning

EXAMPLES A string literal such as **"hello world"** or an open-tag **<** in XML

3.13.1.2
language construct
syntactically allowable part that may be formed from one or more lexical tokens in accordance with the rules of an artificial language

3.13.1.3
predefined identifier
identifier that is defined as part of an artificial language

EXAMPLE A reserved word in a programming language

NOTE If a predefined identifier is not reserved, then a declaration using that identifier redefines its meaning for the scope of the declaration.

3.13.1.4
reserved word
predefined identifier that cannot be redefined

3.13.1.5
delimiter
lexical token that indicates the beginning or the end of another lexical token or of a character string considered as a syntactic unit

NOTE 1 Special characters or reserved words may serve as delimiters.

NOTE 2 Contrast with separator.

3.13.1.6
separator
delimiter that prevents adjacent lexical tokens or syntactic units from being interpreted as a single item

EXAMPLES The space character or a format effector

NOTE Contrast with delimiter.

3.13.1.7
to overload
to assign more than one meaning to a lexical token

EXAMPLE The lexical token "+" can mean integer addition, real addition, set union, concatenation, etc.

3.13.1.8
disambiguation
action of determining which language construct, of several with the same sequence of lexical tokens, is referred to by a particular occurrence

3.13.2 Declarations

3.13.2.1
declaration
explicit language construct that introduces one or more identifiers into a program and specifies how these identifiers are to be interpreted

EXAMPLES Declarations of data types, storage organization, packages, or tasks

NOTE In some programming languages, declarations are considered to be statements.

3.13.2.2
default, adj.
pertaining to an attribute, data value, or option that is assumed when none is explicitly specified

3.13.2.3
scope
scope of a declaration
that portion within which a declaration is valid

3.13.2.4
dynamic scope
scope created by the activation of portions or all of the modules that contain declarations used by another module that lacks these declarations during the execution of the latter module

3.13.2.5
static scope
scope as determined by finding the innermost surrounding module in which the declaration is made

NOTE Static scope may be discovered by visual inspection.

3.13.2.6
local, adj.
pertaining to a language construct that has a scope only within the module in which it is declared

3.13.2.7
global
pertaining to a language construct that is within the scope of all modules

3.13.2.8
external
pertaining to a language construct that is defined outside the module in which it is referenced

NOTE A declaration may be required within the module to provide an identifier and to indicate that the complete definition is external.

3.13.2.9
static
pertaining to objects that exist and retain their values throughout the execution of a processing system

EXAMPLE A subprogram variable that has been declared static to retain its values from one execution to the next

3.14 Terms and definitions from ISO/IEC 2382-17, databases

Definitions are taken verbatim from ISO/IEC 2382-17 unless otherwise stated in a following note.

3.14.1 General terms

3.14.1.1
database
collection of data organized according to a conceptual structure describing the characteristics of these data and the relationships among their corresponding entities, supporting one or more application areas

3.14.1.2
schema
complete description of the structure of a database pertaining to a specific level of consideration

3.14.1.3
data model
description of the organization of data in the management information system of an enterprise

3.14.1.4
data structuring rule
rule that specifies the structure of data as instances of a certain datatype

3.14.1.5
data object
discrete data, considered as a unit, representing an instance of a datatype that is known or assumed to be known

3.14.1.6
data manipulation rule
prescription for manipulating data objects as instances of a certain datatype according to the permissible operations upon data of this datatype

3.14.1.7
database schema
set of various schemas, each of which has the following properties: (1) it pertains to a specific level of consideration of a particular universe of discourse or entity world and to the relevant aspects of an appropriate database; (2) it defines the representation forms for the consistent collection of those sentences of the information base relevant to its respective level of consideration, and it includes the manipulation aspects of these forms

3.14.1.8
database subschema
part of a database schema for one or more applications

3.14.2 Conceptual level

3.14.2.1
entity
concrete or abstract thing that exists, did exist, or might exist, including associations among these things

EXAMPLES A person, an object, an event, an idea, a process, etc.

NOTE An entity exists whether data about it are available or not.

3.14.2.2
attribute
named property of an entity

3.14.2.3
value
<attribute> specific occurrence of an attribute

EXAMPLE "Blue" is an attribute value for the attribute "color"

3.14.2.4
domain
<attribute> set of all possible attribute values

3.14.2.5
attribute class
set of all possible attribute values, corresponding to the same property, of entity occurrences of an entity class

EXAMPLE The name of a column of a relation table can be viewed as the name of an attribute class

NOTE An attribute class must be a subset of the corresponding attribute domain.

3.14.2.6
relationship
<entity> perceived association among objects, entities, among attributes

NOTE A relationship itself may be considered an object, entity, or attribute.

3.14.2.7
dependency
entity relationship or an attribute relationship that denotes that the existence of one entity or attribute is of interest only if another entity or attribute, respectively, exists

3.14.2.8
action
<database> series of insertions, deletions or retrievals of a collection of sentences in an information base or conceptual schema that changes them into a collection of sentences or that makes them known

3.14.2.9
permissible action
action conforming to specific provisions

3.14.3 External, internal (logical and physical)

3.14.3.1
external level
level of consideration at which all aspects deal with the user-oriented representation of information visible at the input and the output of an information system

3.14.3.2
internal level
level of consideration at which all aspects deal with the representation of information within a physical implementation of an information system

3.14.3.3
external schema
part of a database schema that pertains to the external level and that defines the external representations of the possible collections of sentences within a particular user view, including the manipulation aspects of these representations

3.14.3.4
internal schema
part of a database schema that pertains to the internal level and that defines the corresponding representations of the possible collections of sentences within a particular user view, including the manipulation aspects of these representations

3.14.3.5
logical level
level of consideration at which all aspects deal with a database and its architecture, consistent with a conceptual schema and the corresponding information base, but abstract from its physical implementation

3.14.3.6
physical level
level of consideration at which all aspects deal with the physical representation of data structures and with mapping them on corresponding storage organizations and their access operations in a data processing system

3.14.3.7
logical schema
part of the database schema that pertains to the logical level

3.14.3.8
physical schema
part of the database schema that pertains to the physical level

3.14.3.9
storage organization
mapping of a data structure and the operations on its data into a storage device and the corresponding access operations

NOTE The logical elements of the data structure are mapped into their stored physical counterparts; for example, the records of a record type are mapped into stored records of a file.

3.14.3.10
file organization
<entity> arrangement of data in a storage device and the implementation of an access method that are in accordance with the data structures of a particular file and of its records and that provide for the file being part of a database

3.14.3.11
primary key
key that identifies one record

3.14.3.12
secondary key
key that is not a primary key, but for which an index is maintained and that may denote more than one record

3.14.3.13
access path
chain of addresses that leads to the desired data

NOTE There may simultaneously exist more than one access path for one data item.

3.14.3.14
access path independence
separation of a data description from its access path so that changes to the access path do not require changes to the data description in a program

3.14.3.15
current pointer
pointer that is updated, if necessary, at the execution of a data manipulation language statement to identify the location of the current record of the data manipulation

3.14.4 Relational structure

3.14.4.1
relation
set of entity occurrences that have the same attributes, together with these attributes

NOTE In a relational database, a relation can be represented by a table with the rows corresponding to the entity occurrences and the columns corresponding to the attributes.

3.14.4.2
relation class
all relations having identical sets of attributes

NOTE A relation class can be characterized by a set of names of attributes.

3.14.4.3
relational structure
structure of data, in which the data are arranged as relations

3.14.4.4
relational model
data model whose structure is based on a set of relations

EXAMPLE SQL represents such a model

3.14.4.5
relational database
database in which the data are organized according to a relational model

3.14.4.6
relational database management system
RDBMS
database management system designed for relational databases

3.14.4.7
tuple
<relational database> part of a relation that uniquely describes an entity occurrence and its attributes

NOTE A tuple can be represented by one row of a relation table.

3.14.4.8
relational algebra
algebra for expressing and manipulating relations

NOTE Common operations in a relational algebra are projection, selection, join, cartesian product, union, intersection, and difference.

3.14.4.9
projection
operation of relational algebra that forms a new relation by using a subset of the attributes from a given relation

3.14.4.10
selection
operation of relational algebra that forms a new relation which is a subset of the entity occurrences from a given relation

EXAMPLE Given a relation of "books" containing the attributes "author" and "title", the formation of the subset of the books written by a particular author

3.14.4.11
join
operation of relational algebra that forms a new relation from two or more relations having common attribute domains for one or more attributes of each relation

NOTE The operation is based on the Cartesian product of the relations and proceeds by combining rows from the original relations that have identical values from the common domains.

3.14.4.12
normalization
<relational database> process of transforming a relation into one or more simpler relations free of attribute redundancies or inconsistencies in order to support referential integrity

3.14.4.13
referential integrity
property of a set of relations such that the attribute values of foreign keys are null values or are identical to the values of primary keys of other relations

3.14.4.14
cardinality
<relational database> number of tuples in a relation

3.14.4.15
foreign key
in a relation, one or a group of attributes that corresponds to a primary key in another relation

3.14.4.16
cursor
<relational database> pointer to a row in a table, used to move within that table

NOTE In SQL, a current pointer is called a cursor.

3.14.5 Hierarchical and network structures

3.14.5.1
hierarchical model
data model whose pattern of structure is based on a tree structure

3.14.5.2
tree structure
<database> data structure that arranges entities or attributes as nodes, with at most one parent node for each node, and with only one root node

3.14.5.3
network model
data model whose pattern of structure is based on a network structure

EXAMPLE The Network Database Language (NDL) model

3.14.5.4
network structure
data structure that arranges entities or attributes as nodes and that, in contrast to a tree structure, permits nodes to have multiple parent nodes

3.14.5.5
root node
<database> node that has no parent node

3.14.5.6
parent node
<database> node to which at least one other node is directly subordinate

3.14.5.7
terminal node
leaf
<database> node that has no subordinate node

3.14.5.8
record
data object that is an instance of a record datatype

3.14.5.9
realm
<database> part of a database that can be opened and closed as a unit

3.15 Terms and definitions from ISO/IEC 2382-18, distributed data processing

Definitions are taken verbatim from ISO/IEC 2382-18 unless otherwise stated in a following note.

3.15.1 General terms

3.15.1.1
session
all the activities which take place during the establishment, maintenance and release of a connection

3.16 Terms and definitions from ISO/IEC TR 10000-1

The following terms have been incorporated or adapted from ISO/IEC TR 10000-1.

3.16.1 Relationships among normative documents

3.16.1.1
base standard
approved standard used for creating derived standards

[adapted[9] from ISO/IEC TR 10000-1]

3.16.1.2
international standardized profile
ISP
internationally agreed-to, harmonized normative document which describes one or more profiles

3.16.1.3
profile
set of one or more base standards and/or ISPs, and, where applicable, the identification of chosen classes, conforming subsets, options and parameters of those base standards, or ISPs necessary to accomplish a particular function

NOTE ISPs may contain normative references to specifications other than International Standards; see document JTC1/N4047: *The Normative Referencing of Specifications other than International Standards in JTC1 International Standardized Profiles — Guidelines for ISP Submitters*.

[9] The source definition in ISO/IEC TR 10000-1 is particular to JTC1 and ITU-T; the JTC1 and ITU-T references were removed.

3.16.1.4
IT system
set of IT resources providing services at one or more interfaces.

3.16.2 Conformance terminology

3.16.2.1
implementation conformance statement
ICS
statement made by the supplier of an implementation or IT system claimed to conform to one or more specifications, stating which capabilities have been implemented, specifically including the relevant optional capabilities and limits

3.17 Terms and definitions from ISO/IEC 11404

Definitions are taken verbatim from ISO/IEC 11404 unless otherwise stated in a following note.

3.17.1 Fundamental concepts of datatypes

3.17.1.1
datatype
set of distinct values, characterized by properties of those values, and by operations on those values

3.17.1.2
value space
set of values for a given datatype

3.17.1.3
characterizing operations (of a datatype)
collection of operations on, or yielding, values of the datatype that distinguish this datatype from other datatypes with identical value spaces

EXAMPLE The integers may have characterizing operations Add(), Negate(), Multiply(), Quotient(), and Remainder(), while the rational numbers include additional characterizing operations, such as Reciprocal()

3.17.2 Computational aspects of datatypes

3.17.2.1
regular value
element of a value space that is subject to a datatype's properties and characterizing operations

3.17.2.2
sentinel value
element of a value space that is not subject to a datatype's properties and characterizing operations

3.17.2.3
variable
computational object to which a value of a particular datatype is associated at any given time; and to which different values of the same datatype may be associated at different times

3.17.2.4
representation (of a datatype)
mapping from the value space of the datatype to the value space of some internal datatype of a computer system, file system or communications environment

3.17.3 Derived and generated datatypes

3.17.3.1
datatype family
collection of datatypes which have equivalent characterizing operations and relationships, but value spaces that differ in the number and identification of the individual values

3.17.3.2
aggregate datatype
generated datatype each of whose values is made up of values of the component datatypes, in the sense that operations on all component values are meaningful

3.18 Terms and definitions from ISO/IEC 11179

Definitions are taken verbatim from ISO/IEC 11179 unless otherwise stated in a following note.

3.18.1 General terms

3.18.1.1
metadata
data that defines and describes other data or processes

3.18.1.2
metadata registry
MDR
information system for registering metadata

3.18.2 Registry metamodel

3.18.2.1
administered item
registry item for which administrative information is recorded in an administration record

3.18.2.2
attribute
characteristic of an object or entity

3.18.2.3
value domain
VD
set of permissible values

3.18.2.4
enumerated value domain
value domain that is specified by a list of all its permissible values

3.18.2.5
non-enumerated value domain
value domain that is specified by a description rather than a list of all permissible values

3.18.2.6
permissible value
ordered pair consisting of a value and its corresponding value meaning

3.18.2.7
value
data value

3.18.2.8
value meaning
meaning or semantic content of a value

NOTE Given a permissible value, representation of its value meaning shall be independent of (and shall not constrain) the representation of its corresponding value.

3.18.3 Registration

3.18.3.1
registration
relationship between an administered item and the registration authority

3.18.3.2
registration authority
RA
organization responsible for maintaining a registry

3.18.3.3
registration status
designation of the status in the registration life-cycle of an administered item

3.19 Terms and definitions from ISO/IEC 13886

Definitions are taken verbatim from ISO/IEC 13886 unless otherwise stated in a following note.

3.19.1 Parameters

3.19.1.1
client interface binding
possession by the client procedure of an interface reference

3.19.1.2
parameter
used to communicate a value from a client to a server procedure; the value supplied by the client is the actual parameter, the formal parameter is used to identify the received value in the server procedure

3.19.1.3
formal parameter
name symbol of a parameter used in the definition of a procedure to which a value will be bound during execution

3.19.1.4
actual parameter
value that is bound to a formal parameter during the execution of a procedure

3.19.1.5
input parameter
formal parameter with an attribute indicating that the corresponding actual parameter is to be made available to the server procedure on entry from the client procedure

3.19.1.6
output parameter
formal parameter with an attribute indicating that the corresponding actual parameter is to be made available to the client procedure on return from the server procedure

3.19.1.7
input/output parameter
formal parameter with an attribute indicating that the corresponding actual parameters are made available to the server procedure on entry from the client procedure and to the client procedure on return from the server procedure

3.19.2 Procedures and interfaces

3.19.2.1
interface reference
identifier that denotes a particular interface instance

3.19.2.2
marshalling
process of collecting actual parameters, possibly converting them, and assembling them for transfer

3.19.2.3
unmarshalling
process of disassembling the transferred parameters, possibly converting them, for use by the server procedure on invocation or by the client procedure upon procedure return

3.19.2.4
procedure
procedure value

3.19.2.5
procedure value
closed sequence of instructions that is entered from, and returns control to, an external source

3.19.2.6
procedure type
family of datatypes each of whose members is a collection of operations on values of other datatypes

3.19.3 Invocation

3.19.3.1
procedure call
act of invoking a procedure

3.19.3.2
termination
predefined status related to the completion of a procedure call

3.19.3.3
procedure return
act of returning from the server procedure with a specific termination

3.19.3.4
client procedure
sequence of instructions which invokes another procedure

3.19.3.5
server procedure
procedure which is invoked by a procedure call

3.19.3.6
invocation context
for a particular procedure call, instance of the objects referenced by the procedure, where the lifetime of the objects is bounded by the lifetime of the call

3.19.3.7
procedure execution context
for a particular procedure, an instance of the objects satisfying the external references necessary to allow the procedure to operate, where these objects have a lifetime longer than a single call of that procedure

3.19.3.8
procedure image
representation of a value of a particular procedure type, which embodies a particular sequence of instructions to be performed when the procedure is called

3.19.3.9
procedure invocation
object which represents the triple: procedure image, execution context, and invocation context

3.20 Terms and definitions from ISO/IEC 19501

Definitions are taken verbatim from ISO/IEC 19501 unless otherwise stated in a following note.

3.20.1 Fundamental concepts of metamodel

3.20.1.1
association
<UML> definition of a semantic relationship among classifiers

3.20.1.2
association class
<UML> association that is also a class

3.20.1.3
association end
<UML> endpoint of an association, which connects

3.20.1.4
class
description of a set of objects that share the same attributes, operations, methods, relationships, and semantics

3.20.1.5
relationship
<UML> connection among model elements

NOTE In UML, a relationship may be an association, a dependency, a flow, or a generalization.

3.21 Terms and definitions particular to this document

The following terms are particular to this document.

3.21.1 Characteristics of implementations

3.21.1.1
implementation feature
artifact associated with an implementation

3.21.1.2
implementation value
quantifiable artifact associated with an implementation

3.21.1.3
implementation behavior
observable actions or appearance of an implementation

NOTE Implementation behavior may be specified by performance provisions.

3.21.2 Implementation documentation

3.21.2.1
implementation documentation
collection of documents that describe the capabilities, limitations, and other information required for an implementation

EXAMPLE An implementation conformance statement and its supporting documentation

3.21.2.2
application documentation
collection of documents that describe the requirements, capabilities, limitations, design, operation, and maintenance of application software or an application program

EXAMPLE User and installation documentation for a program

3.21.3 Deemed-to-satisfy provisions not necessarily requiring implementation documentation

3.21.3.1
unspecified, adj
<technical specification> lacking of one or more provisions

NOTE 1 The term "unspecified" is always relative in nature, i.e., the determination of "enough provisions" is dependent upon context. Compare to "undefined".

NOTE 2 The term "unspecified" itself does not imply a deemed-to-satisfy provision, but related terms may be deemed-to-satisfy provisions, e.g., unspecified feature, unspecified value, unspecified behavior.

3.21.3.2
unspecified feature
<technical specification> implementation feature, within a content, for which a normative document provides two or more alternatives and imposes no further requirements on what is chosen in any instance

3.21.3.3
unspecified value
<technical specification> implementation value, within a content, for which a normative document provides two or more alternatives and imposes no further requirements on what is chosen in any instance

3.21.3.4
unspecified behavior
<technical specification> implementation behavior, within a content, for which a normative document provides two or more alternatives and imposes no further requirements on what is chosen in any instance

3.21.3.5
smallest permitted maximum
SPM
<technical specification> meet-or-exceed provision for a required minimum value that specifies an implementation value describing the maximum value of a sizing parameter

NOTE An SPM sets a lower bound for an implementation-defined maximum value.

EXAMPLE In the provision "the smallest permitted maximum length of field **X** shall be **17**", the SPM is **17** (which applies to the field length). This provision means: implementers may implement "field **X**" with a maximum length of **17**, **18**, **19**, etc., but not with a length of **16** or less. Thus, `char X[17]` satisfies the implementation requirements, even though the data itself might be smaller, e.g., a 10-character string stored in a 17-character array

3.21.4 Deemed-to-satisfy provisions requiring implementation documentation

3.21.4.1
implementation-defined, adj
<technical specification> unspecified, yet each implementation documents how the choice among the available alternatives is made

NOTE The distinction between "unspecified" and "implementation-defined" is that the latter requires implementation documentation while the former does not require implementation documentation (nor does the former prohibit implementation documentation).

EXAMPLE 1 An implementation-defined feature; an implementation-defined value; an implementation-defined behavior.

EXAMPLE 2 A standard specifies that size of array **X** is implementation-defined with a minimum size of **17**. This provision implies two requirements: (1) the size of the array is greater than or equal to **17**, and (2) the implementation will document the actual size. This example is a meet-or-exceed provision (e.g., a smallest permitted maximum)

3.21.4.2
implementation-defined value
<technical specification> unspecified value for which each implementation documents how the choice among the available alternatives is made

3.21.4.3
implementation-defined behavior
<technical specification> unspecified behavior for which each implementation documents how the choice among the available alternatives is made

EXAMPLE 1 The exception handling associated with integer overflows

EXAMPLE 2 The exception handling with syntax errors on input data; the exception handling associated with file creation errors

3.21.5 Deemed-to-satisfy provisions requiring implementation documentation

3.21.5.1
undefined, adj
<technical specification> absence of requirements and/or description such that meaning is not understood

NOTE 1 The term "undefined" is always relative in nature, i.e., the dependent upon the scope of the context. Compare to "unspecified".

NOTE 2 If there exists any requirements, then the requirements (although possibly incomplete) would be a deemed-to-satisfy provision.

3.21.5.2
undefined feature
<technical specification> implementation feature, within a content, for which a normative document imposes no requirements

3.21.5.3
undefined value
<technical specification> implementation value, within a content, for which a normative document imposes no requirements

3.21.5.4
undefined behavior
<technical specification> implementation behavior for which a normative document imposes no requirements

NOTE Possible undefined behaviors include, but are not limited to: ignoring the situation completely; unpredictable results; behaving in a documented manner characteristic of the environment; terminating processing.

3.21.5.5
defined, adj
<technical specification> sufficient specification of requirements and/or description

NOTE The terms "defined", "unspecified", and "undefined" are related as follows: "defined" indicates completeness in specification, but does not necessarily exclude the possibility of a deemed-to-satisfy provision; "unspecified" indicates a deemed-to-satisfy provision, but does not indicate whether or not the lack of provisions is intended to be complete; "undefined" indicates the lack of description to give sufficient meaning. Thus, a feature may be: defined (but not unspecified), defined and unspecified, or undefined.

3.21.6 Conformance scope

3.21.6.1
in-scope
affirmative sense of applicability with respect to scope

3.21.6.2
out-of-scope (1)
negative sense of applicability with respect to scope

3.21.6.3
out-of-scope (2)
pertaining to features that are not specified in a normative document and may be specified elsewhere

3.21.6.4
scope
defines, without ambiguity, the subject and aspects covered, thereby indicating the limits of applicability [ISO/IEC Directives, Part 2, 5th edition, subclause 6.2.1]

3.21.7 Conformance kinds

NOTE Some terms in this subclause use variants of words that depend upon usage. For example, strictly conform and strictly conforming both designate the same concept, but represents variants of usage "implementation X strictly conforms" or "X is a strictly conforming implementation".

3.21.7.1
conform
conformity
conforming
C
property, with respect to a normative document, that indicates the satisfaction of requirements

3.21.7.2
conformance
act of conforming

NOTE The distinction between "conformance" and "conformity" is the former refers to an action while the latter refers to a state.

3.21.7.3
non-conformity
nonconforming
not conform
not conforming
NC
property, with respect to a normative document, that indicates the lack of satisfaction of requirements

3.21.7.4
non-conformance
act of not conforming

3.21.7.5
strictly conform
strictly conforming
SC
property, with respect to a normative document, that (1) indicates the satisfaction of requirements, (2) indicates, for meet-or-exceed provisions, the absence of implementation features that exceed the minimum requirements, (3) indicates, for provisions other than meet-or-exceed provisions, the absence of implementation--specific features, (4) indicates the absence of unspecified implementation features, and (5) indicates the absence of undefined implementation features

NOTE A strictly conforming implementation implies a conforming implementation, i.e., it is not possibly to be both strictly conforming and non-conforming.

3.21.7.6
strict conformance
act of strictly conforming

3.21.7.7
merely conform
merely conforming
MC
property of conforming, but not strictly conforming

3.21.7.8
mere conformance
act of merely conforming

3.21.7.9
SC/C
strictly conform/conform
strictly conforming/conforming
parallel linguistic construction meaning both "strictly conforming" and "conforming"

NOTE The term "SC/C" functions similar to "and/or" (meaning both "and" and "or").

EXAMPLE 1 The provision "a SC/C implementation shall SC/C to the data model **X**" is equivalent to two provisions: (1) "a strictly conforming implementation shall strictly conform to the data model **X**", and (2) "a conforming implementation shall conform to the data model **X**"

EXAMPLE 2 If record **R** has two mandatory data elements **J** and **K**, and **R** permits extended data elements, then the record **{ J , K }** can be designated SC (strictly conforming), C (conforming), or SC/C (both strictly conforming and conforming), the choice of designation is dependent upon the intent of the conformity statement; but the record **{ J, K }** can be designated C (conforming) or MC (merely conforming)

3.21.8 Conformance framework

3.21.8.1
conformance paradigm
description of the nature and kind of roles of conformance claims

EXAMPLE 1 A communications protocol may have a conformance paradigm that involves **2** roles: a client and a server. In this conformance paradigm, implementations may claim conformance to the client role, the server role, or both

EXAMPLE 2 A data interchange standard may have a conformance paradigm that involves **4** roles: a data instance (i.e., data conforms to the standard), a data reader (e.g., an import tool conforms to the standard), a data writer (e.g., an export tool conforms to the standard), a data repository (e.g., a database that conforms to the standard)

EXAMPLE 3 A programming language standard may have a conformance paradigm that involves **3** roles: a program (that conforms to the language description), a translator (e.g., a compiler that conforms to the standard), an environment (e.g., an operating system conforms by providing services)

NOTE 1 A conformance paradigm is not a deemed-to-satisfy provision because each conformance role of a conformance paradigm represents a *different* set of provisions, not the same set of provisions that may be implemented differently.

NOTE 2 The use of conformance paradigms, typically, simplifies the development, presentation, editing, and maintenance of a normative document. For example, a communications protocol standard with a client-server conformance paradigm could be rewritten as two separate standards, "communications protocol for a server" and "communications protocol for a client", but developing two documents may be more difficult because there may be much normative wording in common between the two documents, and it may be less convenient to read two documents than a single document.

3.21.8.2
conformance role
function performed by an implementation in a conformance paradigm

NOTE An implementation make play more than one conformance role, e.g., in a client-server conformance paradigm, an implementation may be a client, server, or both; in a data interchange conformance paradigm, an implementation may be both a data reader and a data writer (but not a data repository).

3.21.9 Implementation adaptation and specialization

3.21.9.1
locale
common characteristics and their properties for users based upon location

3.21.9.2
locale-specific, adj
changing, dependent upon locale

3.21.9.3
locale-specific feature
implementation-defined feature that changes, depending upon locale

3.21.9.4
locale-specific value
implementation-defined value that changes, depending upon locale

3.21.9.5
locale-specific behavior
implementation-defined behavior that changes, depending upon locale

3.21.9.6
user
human, his/her agent, a surrogate, or an entity that interacts with information processing systems

3.21.9.7
user context
state of a user and his/her surrounding environment

NOTE A user's surrounding environment may include IT components and non-IT components.

3.21.9.8
user-contextualized, adj.
changing, dependent upon user context

NOTE Typically, a user's context includes application-independent data or application-specific data that is user-independent. For example, in a banking application a user's context might include how numbers are displayed (the "thousands separator" and "decimal indicator" are application-independent), the currency notation used (the spelling and placement of currency symbols are application-specific to banking and finance, but are user-independent), but the user's context might exclude the bank balance (application-specific and user-dependent data) because this is merely user data.

3.21.9.9
user-contextualized feature
implementation-defined behavior that changes, depending upon user context

3.21.9.10
user-contextualized value
implementation-defined value that changes, depending upon user context

3.21.9.11
user-contextualized behavior
implementation-defined behavior that changes, depending upon user context

3.21.10 Data associated with implementation adaptation and specialization

3.21.10.1
locale identifier
designation that denotes a locale

NOTE 1 A location may imply cultural, linguistic, or regional conventions.

NOTE 2 See also: localized string, localized value, multicultural value, multicultural string.

EXAMPLE The locale

```
en-GB;GMT0BST;dd.mm.yyyy hh:mm
```

might mean the following conventions apply:

— The language is English, specifically British English.
— The timezone in the winter is called GMT and is 0 hours West of UTC.
— The timezone in the summer is called BST.
— The date and time are written in the format "dd.mm.yyyy hh:mm".

3.21.10.2
localized string
user-context-dependent string determined by a particular locale

3.21.10.3
localized value
user-context-dependent value determined by a particular locale

3.21.10.4
multistring
MUC string
multi-user-context string
set of characterstrings and their expression contexts, that are intended to represent the same meaning, but are expressed differently according to some culture, language, function, or other user context

NOTE 1 User contexts may be distinguished by locales.

NOTE 2 Elements of a multistring may be indexed by locale identifiers.

EXAMPLE 1 The of strings { "standardization", "normalisation", "стандаятизация", "Normungsarbeit", "normalización", "noramzione", "normalisatie", "standardisering" } all have the same meaning (as per ISO/IEC Guide 2)

EXAMPLE 2 A multistring for the concept "July 1, 1999 6PM" (New York City time) might be:

```
multistring_for_july_1 =
(
      ( "en-GB;GMT0BST;dd.mm.yyyy hh:mm",
        "01.07.1999 23:00" )
      ( "en-US;EST5EDT;mm/dd/yyyy hh:mm aa",
        "07/01/1999 06:00 PM" )
      ("iso_8601;UTC0;yyyy-mm-dd hh:mm tz",
        "1999-07-01 22:00 UTC" )
)
```

This multistring `multistring_for_july_1` contains three elements: the localized string for London ("01.07.1999 23:00"), the localized string for New York City ("07/01/1999 06:00 PM"), and the localized string for ISO 8601 ("1999-07-01 22:00 UTC").

EXAMPLE 3 A multistring for the phrase "information technology" might be:

```
mcstring_for_IT =
(
      ( "fr-CA", "Technologies de l'information" ),
      ( "en-US", "Information technology" ),
)
```

This multistring `multistring_for_IT` contains two elements: the localized string for Canadian French, and the localized string for US English.

3.21.10.5
multivalue
MUC value
multi-user-context value
set of values of a particular datatype and their expression contexts, that are intended to represent the same meaning, but are expressed differently according to some culture, language, function, or other user context

NOTE 1 A multivalue is similar to a multistring except that the base datatype of the latter is characterstring while the base datatype of the former is unspecified (by this definition).

NOTE 2 User contexts may be distinguished by locales.

NOTE 3 Elements of a multivalue may be indexed by locale identifiers.

3.21.10.6
context-dependent string
UCD string
user-context-dependent string
element of an multistring

NOTE Context may be dependent upon locale.

3.21.10.7
context-dependent value
UCD value
user-context-dependent value
element of an multivalue

NOTE Context may be dependent upon locale.

3.21.10.8
context-independent string
UCI string
user-context-independent string
string the has the same meaning regardless of culture, language, function, or other user context

EXAMPLE The date "1999-07-01" is context-independent string that represents July 1, 1999. The specification for this string is ISO 8601, Date and Time Formats

NOTE 1 There may be more than one context-independent string that has the same meaning, e.g., "1999-07-01" and "19990701" are both context-independent strings that have the same meaning.

NOTE 2 A context-independent string may be an element of a multistring.

3.21.10.9
context-independent value
UCI value
user-context-independent value
value of a particular datatype the has the same meaning regardless of culture, language, function, or other user context

EXAMPLE The value `"004"` is a context-independent value that represents Afghanistan, as specified by ISO 3166-1, Country Codes

3.21.11 Signifiers and symbols

3.21.11.1
signifier
sign, noun
general concept, whose extension is perceivable objects that are associated with objects

NOTE A signifier associated via a designating relationship to an object of the variety "concept" is known as a signifier designating a concept, i.e., a designation.

3.21.11.2
designation formation
process of creating a designation and associating it with its concept(s)

EXAMPLE 1 An individual or committee that chooses a signifier (of possibly several potential signifiers) to designate a concept

EXAMPLE 2 A registration authority that creates a registration identifier from an object registration process

EXAMPLE 3 A motor vehicle registration service that gives vehicle tags ("license plates") for newly registered vehicles, i.e., the tag (a signifier) designates the registration event (an individual concept)

3.21.11.3
to reference, verb
association with a particular object (the referent)

NOTE In one context, a *reference* is the opposite of a *literal*: the literal gives the data at hand, while the reference points to the data, which must be subsequently accessed, retrieved, or written.

EXAMPLE An association created by proximity; a computer memory pointer; a database foreign key

3.21.11.4
referent
object that is referenced

3.21.11.5
to dereference
to access the referenced object (the referent)

3.21.11.6
label
reference to an object by a signifier

3.21.11.7
label value space
set of possible labels for a naming convention

EXAMPLE In the North American Numbering Plan (the 10-digit telephone numbering scheme for North America), the value space is the set of 10-digit numbers (10,000,000,000 combinations), but not all numbers are *available labels* because a certain subset have been reserved for future use, e.g., 000-000-0000 is an element of the label value space, but not an available label

3.21.11.8
available label
label whose use is permitted

3.21.11.9
label encoding
representation of a label's signifier by one or more codes

EXAMPLE The designation **XYZ123** could be encoded (1) as a series of characters (a text-based encoding), or (2) as a 2-dimensional array of pixels (a graphical-based encoding)

3.21.11.10
label encoding datatype
kind of encoding technique for a label

EXAMPLE Internet URLs are encoded in ASCII as the datatype `characterstring`

3.21.11.11
identifier
label that is intended to be dereferenced

NOTE 1 An identifier is also a reference.

NOTE 2 This definition is consistent with IETF RFC 3986 which describes the Uniform Resource Identifier (URI) syntax and semantics.

3.21.11.12
navigable identifier
identifier that may be used for (an object's) navigation and access

3.21.11.13
identification
associating an identifier with an object

NOTE The process of identification may involve other tasks, such as registration (see ISO/IEC 11179-6).

3.21.11.14
to identify
process of discovering unique or essential characteristics

EXAMPLE 1 An object contains identifiers or has identifiers associated with it. The object is identified by extracting these assigned identifiers from the object

EXAMPLE 2 A criminal may be identified by his/her fingerprints at a crime scene (presumably, fingerprints are unique)

3.21.11.15
identity
collective aspect of the set of characteristics and properties by which a thing is recognizable or known [adapted from Wordnet]

3.21.11.16
locator
identifier that includes an access method

NOTE This definition is consistent with IETF RFC 2396 which distinguishes between a Uniform Resource Identifier (URI) and Uniform Resource Locator (URL).

3.21.11.17
namespace
set of designations with a defined scope of usage

3.21.11.18
namespace label
label that designates a namespace

3.21.12 Data and information

3.21.12.1
value (1)
object whose totality can be used for computation

NOTE In contrast to an object (used in the sense of object-oriented computing), the totality and boundary of a value is known and determinate, whereas the totality or boundary of an (object-oriented) object can be unknown or indeterminate.

3.21.12.2
value (2)
value concept
concept with a defined notion of equality to that concept

NOTE Although any signifier can designate a value, typically representation systems are used to afford computability among signifiers because digital computers are signifier processors, not concept processors. For example, a decimal positional numeration system ($289 = 2×10^2 + 8×10^1 + 9×10^0$) is used by hand calculators and a binary position numeration system ($10001 = 1×2^4 + 0×2^3 + 0×2^2 + 0×2^1 + 1×2^0$) is used by modern digital computers — both position numeration systems afford automated arithmetic computation by employing a pre-determined set of rules for processing signifiers (not processing concepts).

EXAMPLE Given the concepts *red* (defined as "visible light in the range of wavelengths 650 to 700 nanometers") and *yellow* (defined as "visible light in the range of wavelengths 525 to 570 nanometers"), it is undefined whether *red* equals *yellow* and, hence, these concepts by themselves are not values. These concepts can become values once a notion of equality is defined (equality can be defined within the concepts' definition or defined external to the concepts' definition). If the notion of equality is defined as "has visible electromagnetic radiation?", then one can ask whether *red* equals *yellow* (true) and whether *red* equals *infrared* (false, because *infrared* is not visible). If the notion of equality is defined as "has overlapping ranges of wavelengths?", then *red* does not equal *yellow* (their wavelengths don't overlap) but *cornflower* equals *blue* (cornflower is a slice of the blue portion of the visible spectrum). If the notion of equality is defined as "has the same range of wavelengths?", then none of *red*, *yellow*, *cornflower*, or *blue* are equal to each other.

3.21.12.3
datum (pl., data *or* **datums)**
designation whose concept is a value

3.21.12.4
data interoperability
interoperability concerning the creation, meaning, computation, use, transfer, and exchange of data

3.21.12.5
metadata interoperability
interoperability concerning the creation, meaning, computation, use, transfer, and exchange of descriptive data

3.21.12.6
information
given context of an object, such as a concept system, that gives it meaning

NOTE 1 It is possible to give context using techniques other than concept systems.

NOTE 2 A given context might apply to more than one object, such as a class of objects.

NOTE 3 With respect to data and information, a datum already has meaning: its value (concept). Additional meaning might be provided, such as: revealing one or more concept systems that datums belong to (e.g., relations and relationships among data); describing the circumstances of the act of designation (e.g., who-what-when-where-why-how the data was created or changed); providing mappings to/from the designations, their signifiers, and/or their values (concepts). Other methods are possible for giving additional meaning.

NOTE 4 It is possible to provide successive contexts of information for data, each revealing more information, the result of each iteration (information) can itself be considered data for the next iteration of revelation, which produce "layers" of information and data. The reverse process is possible, too: each layer of information is stripped of some context that produces data; then the data itself is treated as information and a second iteration of context is stripped from that information to produce data.

NOTE 5 Because of the successive nature of revelation or stripping, it may appear that terms data and information can be used interchangeably, but this is incorrect. Data is characterized by the signifiers, their associations with concepts, and their notions of equality; information is characterized by referencing the context(s) overlaid upon the data; and both might be present. Likewise, because context can always be added or stripped, it is impossible to say that something is purely data (but no corresponding information) or purely information (but no corresponding data).

EXAMPLE 1 One kind of context is defining the symbols used for communication, e.g., an encoding

EXAMPLE 2 If X is a sequence of bits that represents an encrypted message (the object) whose meaning is merely a sequence of designations whose symbols are zero and one; then the decryption key is an example of context (a mapping) that gives additional meaning to the data to reveal information (i.e., the decoded message)

EXAMPLE 3 The designation "20" is a datum, a kind of object. The context "temperature in degrees Celsius of New York City at 2003-07-19 16:00 UTC" is information about this object

3.21.12.7
metadata
descriptive data about an object

NOTE The term object is defined in x.x.x.x.

3.21.12.8
data instance
element of the value space from a datatype

3.21.12.9
instantiation (of a datatype)
selection of an element of the value space from a datatype

NOTE The instantiation of a datatype may be considered "creating data".

3.21.12.10
instantiation (of a class)
result of invocation of a constructor method

NOTE The instantiation of a class may be considered "creating an object".

3.21.12.11
pointer
<organization of data> datatype for referencing an object

3.21.12.12
characterstring
datatype that describes a finite sequence of characters from the repertoire of a particular character set

EXAMPLES An ASCII characterstring; an ISO/IEC 10646 characterstring

3.21.12.13
list
datatype containing finite, ordered set of related items

NOTE The items in a list may be lists themselves.

3.21.12.14
aggregate
<datatype> instance of an aggregate datatype

3.21.12.15
data structure
an instance of an aggregate datatype of zero or more components

EXAMPLES A record; a set; a sequence; a list; an array

NOTE A data structure itself may be a data element in a larger data structure.

3.21.12.16
logical record
record whose data elements are related from a logical viewpoint, independent of their physical environment

NOTE 1 Portions of one logical record may be located in different physical records, or several logical records or parts of logical records may be located in one physical record.

NOTE 2 Metadata may describe data, data elements, or other objects.

NOTE 3 Metadata may include data descriptions, data about data ownership, access paths, access rights, and data volatility.

3.21.12.17
physical record
record located in one physical position on a data medium or in a storage device

3.21.12.18
record length
record size
number of bytes, or any other appropriate unit, in a record

3.21.12.19
data repository
<data> implementation of a collection of data along with data access and control mechanisms, such as search, indexing, storage, retrieval and security

EXAMPLE A repository might support services such as search, indexing, storage, retrieval and security

3.21.12.20
resource
anything that has identity

[ISO 15836 and IETF RFC 2396]

EXAMPLES An electronic document, an image, a service (e.g., "today's weather report for Los Angeles"), and a collection of other resources. Not all resources are "retrievable" via computer networks; e.g., human beings, corporations, and bound books in a library can also be considered resources

3.21.12.21
semi-structured data
aggregate datatype whose components' datatypes and their labels are not predetermined

3.21.12.22
code, verb
process of representing information in some structure

3.21.12.23
coding, noun
formalized or structured representation of information

3.21.12.24
encoding
coding that maps to a sequence of bits and/or bytes

NOTE 1 The result of encryption is ciphertext.

NOTE 2 The reverse process is called decryption.

3.21.13 Conformance framework for data interchange

3.21.13.1
data interchange
data exchange
concerning the representation, transmission, reception, storage, and retrieval of data

NOTE In reference to the terminology of ISO/IEC 2382-1, *data interchange* is *information processing* excluding *data processing* (e.g., excluding the operational aspects involving humans).

3.21.13.2
data interchange conformance paradigm
conformance paradigm that includes the roles: data instance, data reader, data writer, and data repository

NOTE Normative documents that specify a data interchange conformance paradigm may include additional conformance roles.

3.21.13.3
transformation phase
<data handling> single phase within a series of phases of data processing whereby data is received (e.g., phase N-1), processed (phase N), and emitted for further processing (e.g., phase N+1)

3.21.13.4
data instance
data instantiation
instance (of a datatype)
instantiation (of a datatype)
selection of an element of the value space from a datatype

NOTE The instantiation of a datatype may be considered "creating data".

3.21.13.5
binding
result of applying or mapping one framework or specification to another

3.21.13.6
bound data instance
bound data
data instantiated from a datatype and rendered in a binding

3.21.13.7
data structure
bound data instance of an aggregate datatype

3.21.13.8
data application
<conformance paradigm> implementation of a functional unit interface of a normative document that contains a data interchange specification

EXAMPLES A data reader, a data writer; a data repository

NOTE 1 A functional unit interface may be described by a coding specification, an API specification, a protocol specification, or some other interface specification.

NOTE 2 Given a data interchange specification named **X**, a "**X** data application" is different from a "conforming **X** application". The latter is any information technology implementation that conforms to the **X** specification, while the former is a limited subset of information technology systems, e.g., a data reader, a data writer, a data repository.

3.21.13.9
to consume
to consume data
data consumer
data consumption
<data> to read data and then process it to the extent that lexical or coding boundaries are discovered

NOTE 1 A data consumer performs a limited number of transformation phases. Data is consumed before it is interpreted. See example in definition of "to interpret".

NOTE 2 See also: generate; interpret; produce.

3.21.13.10
to interpret
interpret data
data interpreter
data interpretation
<data> to process data to discover its meaning to the extent required by an application

NOTE 1 In terms of transformation phases, data is consumed before it is interpreted.

EXAMPLE In the following character stream:

```
<R>
  <A>123.45</A>
  <B>PQR</B>
  <C X="Y">Z</C>
</R>
<R>
  <D>JKL</D>
  <E>
    <F>XXX</F>
    <G>YYY</G>
  </E>
</R>
```

a data consumer might recognize: there are two records, both with tags `"R"`; the first `"R"` record contains three records with tags `"A"`, `"B"`, `"C"`; the second `"R"` record contains two records with tags `"D"` and `"E"`. Because only these tags are recognized, only these tags are candidates for data interpretation. Assuming tag `"E"` represents an extended data element, a data interpreter might only recognize the standardized tags `"A"`, `"B"`, `"C"`, and `"D"`. Based on (1) the separation of the "consume" and "interpret" transformation phases, and (2) a particular standards binding (XML-like in this case), an application might only interpret the standardized features `A`, `B`, `C`, and `D`.

NOTE 2 An application that combines data consumption and data interpretation, but only interprets standardized data elements, might be strictly conforming data reader.

NOTE 3 See also: consume; generate; produce.

3.21.13.11
to generate
generate data
data generator
data generation
<data> to transform data from its meaning to some coding suitable for data interchange

NOTE 1 In terms of transformation phases, data is generated before it is produced.

NOTE 2 See also: consume; interpret; produce.

EXAMPLE To serialize a data structure according to a conceptual model without rendering the data in a specific coding or encoding

3.21.13.12
to produce
produce data
data producer
data production
<data> to process data to the extent that lexical or coding boundaries are defined and then to output the resultant data

NOTE 1 Data is generated before it is produced.

NOTE 2 See also: consume; generate; interpret.

3.21.13.13
data reader
data application that operates *as if* it processes data in two transformation phases (1) by consuming data based on the normative document and the application's inputs, and (2) by interpreting the result based on the normative document and creating data instances which are transformed to data that is internal to the application

EXAMPLE An application *consumes* data from an input stream according to specified lexical and syntactic rules; then the application *interprets* the result by specified semantic and deserialization rules to produce an object for use by the application

NOTE 1 The "as if" provision implies that, conceptually, the data reader processes the information in two phases: consumption and interpretation. However, the design of implementations is not constrained; implementations may use any number of phases of data processing.

NOTE 2 A particular data application implies a particular normative document. See definition of "data application".

3.21.13.14
data writer
data application that operates *as if* it processes data in two transformation phases (1) by generating data from application data based on the normative document, and (2) by producing data to the application's outputs based on the normative document

EXAMPLE An application *generates* data by creating objects according to semantic rules; then the application *produces* data by serializing the objects according to syntactic and lexical rules resulting in an output stream

NOTE 1 The "as if" provision implies that, conceptually, the data writer processes the information in two phases: generation and production. However, the design of implementations is not constrained; implementations may use any number of phases of data processing.

NOTE 2 A particular data application implies a particular normative document. See definition of "data application".

3.21.13.15
data repository
functional unit that stores and retrieves data

EXAMPLE A data repository might support services such as search, indexing, storage, retrieval, and security

3.21.13.16
API application
<conformance paradigm> implementation that uses the services and resources of an API specification

EXAMPLE Implementation P, a software program, conforms as an API application because it uses (in contrast to implements) the API according to the requirements of that API standard

3.21.13.17
API environment
<conformance paradigm> implementation that implements the interface, services, resources, etc. of an API specification

EXAMPLE Implementation L, a software library, conforms as an API environment because it implements (in contrast to uses) the interface, services, resources, etc. of the API according to the requirements of the API specification

NOTE Some API environments may be nested in nature and, therefore, the implementation may be both an API environment and an API application.

3.21.13.18
server service
<conformance paradigm> implementation that implements the server portion of a client-server protocol specification

EXAMPLE Implementation S, a software program, conforms as a server service because it implements the server portion of the protocol according to the requirements of that protocol standard

NOTE Typically, server services respond to requests from client services and, possibly, peer services acting as clients.

3.21.13.19
client service
<conformance paradigm> implementation that implements the client portion of a client-server protocol specification

EXAMPLE Implementation C, a software program, conforms as a client service because it implements the client portion of the protocol according to the requirements of that protocol's specification

NOTE Typically, client services make requests to server services and, possibly, peer services that act as servers.

3.21.13.20
peer service
<conformance paradigm> implementation that implements a peer-to-peer protocol specification

EXAMPLE Implementation P, a software program, conforms as a peer service because it implements the peer-to-peer protocol according to the requirements of that protocol's specification

NOTE Typically, peers interact with other peers. Peers may interact with clients and servers when they can act in the role of servers and clients, respectively.

3.21.14 Data element obligation

3.21.14.1
obligation
<data element> requirements and permissibility of components of an aggregate datatype that determine the validity of an instance of the datatype

NOTE 1 Obligation attributes are independent of longevity attributes.

NOTE 2 See also: conditional data element; data element longevity; extended data element; mandatory data element; optional data element.

EXAMPLE A data structure **x**, has four elements: **A** and **B** are mandatory, **C** is optional, and **D** is conditional if **B** has the value true. The following are sample valid and invalid data structures:

```
( A=123 )               // invalid: missing mandatory element B
( A=123, B=false )      // valid
( A=123, B=true )       // invalid: missing conditional element D
( A=123, B=true, D=17 ) // valid
( A=123, B=false, D=17 ) // valid: allowable because the example
                        // "conditional" wording above only
                        // makes requirements and makes no
                        // prohibitions
( A=123, B=nil, C=345 ) // valid
```

3.21.14.2
mandatory data element
component of an aggregate datatype that is defined and is required to exist within an instance of that datatype

NOTE The "mandatory" nature of a data element is an obligation attribute.

3.21.14.3
optional data element
component of an aggregate datatype that is defined and is permitted, but not required, to exist within an instance of that datatype

NOTE 1 The "optional" nature of a data element is an obligation attribute

NOTE 2 The "optional" nature of data element is independent of its longevity, e.g., there may be obsolete optional data elements, provisional optional data elements, reserved optional data elements, and obsolete optional data elements.

3.21.14.4
conditional data element
component of an aggregate datatype that is defined and is required to exist within an instance of that datatype only under certain conditions

NOTE 1 The "conditional" nature of a data element is an obligation attribute.

NOTE 2 See also: extended data element; mandatory data element; obligation; optional data element.

3.21.14.5
extended data element
component of an aggregate datatype that is defined outside the base normative document that specifies the datatype

NOTE 1 The "extended" nature of a data element is an obligation attribute. Depending upon the data interchange agreements, the inclusion of an extended data element in the data instance may be required, permitted, prohibited, or have some other obligation.

NOTE 2 Strictly conforming implementations of a normative document for a data instance are prohibited from including extended data elements, i.e., the inclusion of an extended data element implies that the data instance is conforming at best (there may be other factors that cause the data instance to be non-conforming).

3.21.15 Data element longevity

3.21.15.1
longevity
<data element> attribute of a data element in a data structure that indicates intention for incorporation into past, present, or future editions of the normative document that specifies the data structure

NOTE 1 Longevity attributes are independent of obligation attributes.

NOTE 2 Longevity may be tied to registration status for registry-based data elements.

3.21.15.2
reserved data element
component of an aggregate datatype that is not defined and is required not to exist within an instance of that datatype

NOTE 1 The "reserved" nature of a data element is a longevity attribute.

NOTE 2 A reserved data element may be overridden by the specification of an extended data element. A data element that is intended to preclude use and to preclude being overridden may be accomplished by the use of an optional data element of datatype `void`, i.e., a `void` datatype cannot be overridden by an extended data element.

NOTE 3 Because "reserved" implies that a data element is not defined, it is not possible to have reserved mandatory data elements, reserved optional data elements, reserved conditional data elements, and reserved extended data elements.

3.21.15.3
provisional data element
component of an aggregate datatype that is defined but is for trial use

NOTE 1 The "provisional" nature of a data element is a longevity attribute.

NOTE 2 A prestandard may include provisional data elements.

NOTE 3 Implementers may use provisional data elements with same caution that they use and apply prestandards: provisions may change based upon experience gained by usage.

NOTE 4 The "provisional" nature of data element is independent of its obligation, e.g., there may be provisional mandatory data elements, provisional optional data elements, provisional conditional data elements, and provisional extended data elements.

3.21.15.4
obsolete data element
component of an aggregate datatype that is defined but is no longer in use

NOTE 1 The "obsolete" nature of a data element is a longevity attribute.

NOTE 2 The term "obsolete" differs from the term "deprecated" in that the former concerns the lack of use while the latter is a recommendation against usage regardless reason (e.g., certain features may be deprecated not because of obsolescence, but because of incompatibility, imprecision, or specification error). Typically, the use of obsolete features is deprecated in standards because standards are continually revised to reflect current usage. Thus, obsolete features may be removed from future versions of standards, which might cause interoperability and compatibility problems for implementations that continue to use obsolete features.

NOTE 3 The "obsolete" nature of data element is independent of its obligation, e.g., there may be obsolete mandatory data elements, obsolete optional data elements, obsolete conditional data elements, and obsolete extended data elements.

3.21.16 Relationships among normative documents

3.21.16.1
base normative document
normative document used for creating derived normative documents

3.21.16.2
derived normative document
normative document that has one or more provisions in common with another normative document

3.21.16.3
base technical specification
normative document used for creating derived technical specifications

3.21.16.4
derived technical specification
technical specification that has one or more provisions in common with another normative document

3.21.16.5
derived standard
standard that has one or more provisions in common with another normative document

3.21.17 Extensions and their normative documents

3.21.17.1
base data model
data model that may be used for derivation

EXAMPLE 1 A base data model, plus extensions, forms a new data model (which is derived from the base data model)

EXAMPLE 2 A profile of a base data model forms a new data model (which is derived from the base data model)

EXAMPLE 3 An API binding may be based upon a data model (i.e., the API is derived from the base data model)

NOTE The term _base data model_ is used throughout the ISO/IEC 20944 series of International Standards to reference the implied application's data model that is being used for the bindings. The _base data model_ is tied to the bindings via normative reference, e.g., some other standard defines a data model and uses ISO/IEC 20944, via normative reference, to provide some coding, API, or protocol bindings. For ISO/IEC 20944-5, the "base data model" is the ISO/IEC 11179-3 metamodel.

3.21.17.2
referenced data interchange specification
data model that is being used for a defined interoperability binding

NOTE The term _referenced data interchange specification_ is used throughout the ISO/IEC 20944 series of International Standards to reference the data model that is being used for the bindings. The _referenced data interchange specification_ is tied to the bindings via normative reference, e.g., some other standard defines a data model and uses ISO/IEC 20944, via normative reference, to provide some coding, API, or protocol bindings. For ISO/IEC 20944-5, the _referenced data interchange specification_ refers to the ISO/IEC 11179-3 metamodel.

3.21.17.3
extension
<data interchange> additional provisions with respect to a base normative document

NOTE Extensions may produce a more expansive normative document, a more restrictive normative document, or both.

3.21.17.4
subset normative document
subset technical specification
subset standard
<data interchange> derived normative document whose set of conforming data instances is a subset of the set of conforming data instances of a base normative document that concerns data interchange

3.21.17.5
superset normative document
superset technical specification
superset standard
<data interchange> derived normative document whose set of conforming data instances is a superset of the set of conforming data instances of a base normative document that concerns data interchange

3.21.17.6
coding-independent representation
CIR
representation of data that is independent of storage representation

3.21.17.7
coding-specific representation
CSR
representation of data based upon a defined storage representation

3.21.18 Relations among value spaces and value domains

3.21.18.1
extended value space
value space with addition elements, yet retains the same characterizing operations and datatype properties

3.21.18.2
isomorphic value domains
value domains such that there is a one-to-one and onto equivalence mapping of corresponding value meanings

EXAMPLE Using the 2-letter, 3-letter, and 3-digit country codes from ISO 3166-1, the three value domains whose values are { **CN**, **FR**, **US** }, { **CHN**, **FRA**, **USA** }, { **156**, **250**, **840** } isomorphic to each other because their corresponding value meanings are the same { **"China"**, **"France"**, **"United States"** }

3.21.18.3
overlapping value domains
value domains such that for all values in common, their corresponding meanings are the same

EXAMPLE The value domains { **red**, **orange**, **yellow**, **green** } and { **red**, **yellow**, **green**, **blue** } are overlapping because the values **red**, **yellow**, and **blue** and their corresponding value meanings are the same

3.21.19 Relationships among value domains

3.21.19.1
base value domain
value domain used for creating derived normative documents

3.21.19.2
derived value domain
value domain that has one or more values in common with another value domain

NOTE The values may or may not retain same meaning in the derived value domain. For example, if { **single**, **married** } is used to derive { **single**, **married**, **widowed**, **divorced** }, then the meaning of **single** changes in the derived value domain; if { **male**, **female** } is used to derive { **not known**, **male**, **female**, **not specified** }, then the meanings of **male** and **female** are unchanged in the derived value domain.

3.21.19.3
extended value domain
value domain that is a superset of a base value domain and whose permissible values are retained

NOTE The values and their value meanings of the base value domain are retained in the extended value domain.

3.21.20 Roles of organizations, entities, and individuals

3.21.20.1
administrator
person responsible for certain business operations in an organization

3.21.20.2
role-based access control
RBAC
security technique for authentication that authorizes operations or allows access to resources based upon the user's identity and his/her relationship to other users and entities

EXAMPLE 1 A teacher has read/write access to the grades for his/her students (role: "the teacher of the student"), but no access to other students' grades

EXAMPLE 2 A principal has read-only access to the grades of all of his/her teachers' students (role: "the principal of the teachers of the students"), but the principal is not permitted to change any grades

3.21.21 Miscellaneous

3.21.21.1
asynchronous
property of a process that lacks of specification of timing relationships

NOTE A data communication protocol may be modelled as a process.

3.21.21.2
application area
industry, market segment, or stakeholder class for which a set of related applications are developed

3.21.21.3
child element
<XML> non-root element C in an XML document for which there is one other element P in the document such that C is in the content of P, but C is not in the content of any other element that is in the content of P

NOTE C is referred to as a child of P.

3.21.21.4
file updating
activity of adding, deleting, or changing data in a file

3.21.21.5
parent element
<XML> element P in relationship to element C such that C is a child element of P

NOTE P is referred to as the parent of C.

3.21.21.6
root element
<XML> element such that no part of it appears in the content of any other element within an XML document

NOTE For all other elements, if the start tag is in the content of another element, then the end tag is in the content of the same element. More simply stated, the elements, each delimited by a start tag and end tag, nest properly within each other.

3.21.21.7
software package
complete and documented set of programs supplied to several users for a generic application or function

NOTE Some software packages are alterable for a specific application.

NOTE Although operating systems are predominantly software, partial hardware implementations are possible.

3.21.21.8
wildcard pattern
technique of describing selection criteria by means of a pattern

NOTE A wildcard pattern may be an Extended Backus Naur Form (EBNF) production rule definitions list (i.e., the right-hand side of the production rule) that contains no non-terminals, no defining operators, no comments, and no rule terminators.

EXAMPLE The following wildcard pattern, as described in ISO/IEC 14977 (EBNF):

```
"ab", { ? any ISO/IEC 10646 character ? }
```

matches any string beginning with `"ab"`, such as `"ab"`, `"abc"`, `"abd"`, `"ababab"`, but not `"a"` or `"ba"`. The pattern is decomposed into (1) the terminal string `"ab"`, (2) the concatenation character `","`, (3) the special sequence `"? any ISO/IEC 10646 character ?"`, and (4) the repeat-zero-or-more-times `"{ ... }"` grouping.

3.22 Acronyms, initialisms, and abbreviations

3.22.1
11179
abbreviation for "the ISO/IEC 11179 series of International Standards" (*Information technology — Metadata registries (MDR)*)

NOTE 1 As of the time of publication of this International Standard, ISO/IEC 11179 consists of 6 parts.

NOTE 2 The abbreviation 11179 may be suffixed with a hyphen and a number that indicates a particular part of ISO/IEC 11179, e.g., 11179-3 means ISO/IEC 11179-3.

3.22.2
20943
abbreviation for "the ISO/IEC 20943 series of International Standards" (*Information technology — Procedures for achieving metadata registry content consistency*)

NOTE At of the time of publication of this International Standard, ISO/IEC 20943 consists of 2 parts.

3.22.3
20944
abbreviation for "the ISO/IEC 20944 series of International Standards" (*Information technology — Metadata Registries Interoperability and Bindings (MDR-IB)*)

NOTE See the Foreword and the Introduction for a list of parts in this International Standard.

3.22.4
ASN.1
abbreviation for "the ISO/IEC 8824 and ISO/IEC 8825 series of International Standards" (*Information technology — Abstract Syntax Notation One (ASN.1)* and *Information technology — ASN.1 encoding rules*)

3.22.5
C
C programming language

NOTE See ISO/IEC 9899, *Programming languages — C.*

3.22.6
C++
C++ programming language

NOTE See ISO/IEC 14882, *Information technology — Programming languages — C++.*

3.22.7
ECMAScript
ECMAScript programming language

NOTE See ISO/IEC 16262, *Information technology — Programming languages, their environments and system software interfaces — ECMAScript language specification.*

3.22.8
Java
Java programming language

3.22.9
JMS
Java messaging services

3.22.10
LDAP
lightweight directory access protocol

3.22.11
LISP
LISP programming language

NOTE See ISO/IEC 13816, *Information technology — Programming languages, their environments and system software interfaces — Programming language ISLISP*.

3.22.12
MDR-IB
metadata registry interoperability and bindings

3.22.13
MDR
metadata registry(ies)

3.22.14
Perl
Perl programming language

3.22.15
PHP
PHP programming language

3.22.16
SOAP
simple object access protocol

3.22.17
WSDL
web services description language

3.22.18
XML
extensible markup language

4 Conformance

In this International Standard, "shall" is to be interpreted as a requirement upon an implementation; "shall not" is to be interpreted as a prohibition. Undefined behavior is:

1. a "shall" requirement or "shall not" prohibition is violated
2. indicated in this International Standard by the words "undefined behavior"
3. indicated by the omission of any explicit definition of behavior.

There is no difference in emphasis among these three; they all describe "behavior that is undefined".

NOTE Implementations claim conformance to particular features of this International Standard in their Implementation Conformance Statement (ICS).

4.1 Conformance level

The following subclauses define strictly conforming implementations and conforming implementations. In the context of conformance, the terms "support", "use", "test", "access", and "probe" are defined in each data conformance paradigm that incorporates this Part.

4.2 Profiles, derived standards, subset standards, superset standards, and extensions

Implementations shall indicate which Parts of this International Standard they claim conformity to in their ICS and in their conformance label(s).

NOTE Implementations may use automated techniques to convey their ICS for automated interoperability.

4.3 Strictly conforming implementations

A strictly conforming implementation:

— shall support all mandatory and optional data elements;

— shall not use, test, access, or probe for any extension features;[10]

— shall not exceed limits or smallest permitted maximum values specified by the controlling normative document; and

— shall not interpret or generate data elements that are dependent on any unspecified, undefined, implementation-defined, or locale-specific behavior.

NOTE The use of extensions is undefined behavior.

4.4 Conforming implementations

A conforming implementation shall be at least one of: a conforming coding, a conforming API, a conforming protocol, or a conforming data application.

A conforming implementation:

— shall support all mandatory and optional data elements;

— may use, test, access, or probe for extension features, as permitted by the implementation and data interchange participants, as long as the meaning and behavior of strictly conforming implementations remain unchanged;

— shall not support or use extension features that change the meaning or behavior of strictly conforming implementations;

— may exceed limits or smallest permitted maximum values specified by controlling normative document(s), and to the extent permitted by the implementation; and

— may interpret or generate data elements that are dependent on implementation-defined, locale-specific, or unspecified behavior.

NOTE 1 The use of extensions is undefined behavior.

NOTE 2 All strictly conforming implementations are also conforming implementations.

NOTE 3 An implementation does not conform to this Part if it redefines features via extension methods, and these features change the meaning or behavior of strictly conforming implementations.

[10] Extension features include extended data elements, extended value spaces, extended value domains, extended operations, etc..

4.5 Conformance labels

A conformance label may summarize ICSs. Annex C, Conformance Labels, describes the requirements for human-readable and machine-readable conformance labels.

5 Derived normative documents, profiles, extensions

5.1 Derived normative document (derived standard)

A derived normative document is a normative document that has provisions in common with a base normative document. In comparison to profiles, a derived standard makes no requirements concerned the relationship between conformance to the base normative document and conformance to the derived normative document (or vice versa). Thus an implementation that conforms to the derived normative document is not required to conform to the base normative document (and vice versa). A derived normative document may contain extensions to the base normative document.

5.2 General principles of a profile

A *profile* is defined in ISO/IEC TR 10000-1 as:

> [a] set of one or more base standards and/or ISPs, and, where applicable, the identification of chosen classes, conforming subsets, options and parameters of those base standards[11], or ISPs necessary to accomplish a particular function

Profiles reference other normative documents. References may be dated or undated.

The general principles of a profile are specified in ISO/IEC TR 10000-1:1998, subclause 6.3.1:

> **6.3.1 General Principles**
>
> A profile makes explicit the relationships within a set of base standards used together (relationships which can be implicit in the definitions of the base standards themselves), and may also specify particular details of each base standard being used. A profile may refer to other International Standardized Profiles[12] in order to make use of the functions and interfaces already defined by them, and thus limit its own direct reference to base standards. It follows that a profile
>
> a) shall restrict the choice of base standard options to the extent necessary to maximise the probability of achieving the objective of the profile; for example to facilitate interworking between IT systems, or porting an application between them, where they have implemented different selections of options of the profile. Thus a profile may retain base standard options as options of the profile provided that they do not affect interworking or portability.
>
> b) shall not specify any requirements that would contradict or cause non-conformance to the base standards to which it refers;
>
> c) may contain conformance requirements which are more specific and limited in scope than those of the base standards to which it refers. Whilst the capabilities and behaviour specified in a profile will always be valid in terms of the base standards, a profile may exclude some valid optional capabilities and optional behaviour permitted in those base standards.

[11] Generically, this International Standard refers to *normative documents* and *technical specifications*; a *standard* is a *normative document* that is a *technical specification* and agreed upon by *consensus*. Thus, the intention of using the terms *normative document* and *technical specification* is to decouple the *consensus process* that is associated with a *standard*.

[12] An International Standardized Profile (ISP) is a *profile* that is also an *international standard*.