ISO/IEC JTC1 SC36 N0646

ISO/IEC JTC1 SC36
Information Technology for Learning, Education, and Training

Title:

Proposed Draft Technical Report for: ISO/IEC xxxxx, Information technology -- Learning, education, and training -- Management and delivery -- Specification and use of extensions and profiles

Source:

SC36 Secretariat

Project:

NA

Document Type:

PDTR

Status:

FYI. Also known as WG4/N0070.

Date:

2003-11-13

Action ID:

LB. Ballots are due by 2004-02-13 to the SC36 Secretariat ("sc36-sec@jtc1sc36.org"). Please use the balloting form for your response:
http://jtc1sc36.org/doc/36N0646_ballot_return.doc

Distribution:

P, O, & L Members, WG Conveners

ISO/IEC JTC1 SC36 **WG4 N0070**

ISO/IEC JTC1 SC36
Information Technology for Learning, Education, and Training
ISO/IEC JTC1 SC36 WG4
Management and Delivery

Title:

Proposed Draft Technical Report for: Information technology -- Learning, education, and training -- Management and delivery -- Specification and use of extensions and profiles

Source:

SC36 Secretariat

Project:

NA

Document Type:

Working Document

Status:

FYI. Also known as SC36/N0646.

Date:

2003-11-13

Action ID:

FYI

Distribution:

P, O, & L Members, WG Participants

Reference number of working document: **ISO/IEC JTC1 SC36 WG4 N0070**

Date: 2003-11-13

Reference number of document: **ISO/IEC PDTR1 xxxxx**

Committee identification: ISO/IEC JTC1 SC36 WG4

SC36 Secretariat: US

Information technology — Learning, education, and training — Management and delivery — Specification and use of extensions and profiles

Warning

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as a Technical Report.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type: Technical Report
Document subtype: if applicable
Document stage: (30) Committee
Document language: E

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member 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 shall not be held responsible for identifying any or all such patent rights.

ISO/IEC TR xxxxx was prepared by Technical Committee ISO/IEC JTC1, *Information Technology*, Subcommittee SC36, *Information Technology for Learning, Education, and Training*.

Introduction

This document was developed to review and analyze the use of "extensions" and "profiles" of data interchange standards and specifications. This document addresses the following points from both theoretical and practical perspectives:

- Presentation of core terminology, compiled from several sources.
- What is a normative document? a technical specification? a standard?
- What are data interchange standards/specifications?
- What issues arise when standards are adopted?
- Conformity, conformance
- What is an extension?
- What issues arise when using extensions?
- What is a profile? What is a derived standard?
- What issues arise when using profiles and derived standards?
- Comparison of extensions vs. profiles
- Conclusions and recommendations

This work builds upon the work product of several standards activities, such as

- ISO/IEC, developers of ISO/IEC Guide 2
- ISO/IEC JTC1 — Information Technology, developers of ISO/IEC 2382 and ISO/IEC TR 10000
- ISO/IEC JTC1 SC22 — Programming Languages, Their Environments, and System Software Interfaces, developers of ISO/IEC 11404
- ISO/IEC JTC1 SC32 — Data Management and Interchange, developers of ISO/IEC 11179 and ISO/IEC 20944
- ISO/IEC JTC1 SC36 — Information Technology for Learning, Education, and Training
- ISO/IEC JTC1 SC36 WG4 — Management and Delivery

This work incorporates several extensive examples, including the work product from the Canada-Finland sponsored survey of metadata implementations (ISO/IEC JTC1 SC36 WG4 N0057) and prior work from IEEE 1484.14 Semantics and Exchange Bindings WG.

The authors thank the above-named committees and their participants for their valuable work.

NOTE This document includes excerpts from ISO/IEC Guide 2, ISO/IEC 2382, ISO/IEC TR 10000, ISO/IEC 11404, ISO/IEC 11179, and ISO/IEC 20944.

Information technology — Learning, education, and training — Management and delivery — Specification and use of extensions and profiles

1 Scope

This document presents common requirements, issues, hazards, and solutions concerning the medium-term and long-term development and adoption of data interchange standards. The development of a data interchange standard (or specification) is a complex task involving competing stakeholder needs. Some of these needs can be characterized by several dichotomies, such as:

— **Codify only existing practice vs. develop a complete solution:** Existing practice has the advantage that some implementations are feasible and immediately available (but widespread interoperability and implementation might not be confirmed). A complete solution may have better interoperability, feasibility, and adoption (but the complete solution may require significant time/effort to develop).

— **A "partially-baked" specification now vs. a "fully-baked" specification N months/years from now:** An immediate solution may satisfy some stakeholders immediate needs (but may be a poor choice for the medium-term and long-term). A fully developed solution will be more completed and coherent (but might miss a "window of opportunity").

— **More functionality vs. lower implementation cost:** Some stakeholders want substantially more functionality (e.g., users), while other stakeholders want to minimize the development cost of the implementation (e.g., vendors).

— **Single phase delivery vs. multiple phase delivery:** The delivery of a single specification will be better integrated (but it would prohibit early delivery of stable or independent specification modules). The delivery of multiple specification modules will make components available when ready (but may causes revisions as later modules affect and update earlier-delivered specification modules).

— **Backward compatibility vs. future capability:** Prior systems and versions must be supported so that existing systems do not have to change (but old technologies, methodologies, and approaches may limit the ability to satisfy current or future needs). Solutions that address current and future needs may be the most appropriate (but may limit the interoperability with existing or legacy systems).

— **General solution vs. application-specific solution:** A general solution is widely applicable (but may require significant analysis, time, and effort to develop). An application-specific solution is closely matched to a particular application's needs (but may be less interoperable and may be less adaptable to current or future needs).

Stakeholders continually address these issues, although it is can be a difficult balancing act — and the outcomes are likely to be different for each data interchange standard.

This document is intended to be a useful reference for the stakeholders, such as users, vendors, and organizations.

2 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 Directives, Part 2, 4th edition, *Rules for the structure and drafting of International Standards*

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

ISO/IEC 2382 (parts 1, 4, 17), *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 11179 (parts 1, 3, 6), *Information technology — Metadata Registries (MDR)*

ISO/IEC CD 20944-002, *Information technology — Metadata Interoperability and Bindings (MDIB) — Common Vocabulary*

ISO/IEC JTC1 SC36 WG4 N0057, *Survey of LOM Implementations: Preliminary Report* (N. Friesen, L. Nirhamo, "http://mdlet.jtc1sc36.org/doc/SC36_WG4_N0057.pdf")

3 Terms and definitions

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

NOTE The following terminology has been compiled from several sources. The purpose of this compilation is to provide the reader with all the relevant terms in a single document so that the remainder of the document can be presented both concisely and precisely.

3.1 Terms and definitions from ISO/IEC Guide 2

The following terms have been incorporated or adapted from ISO/IEC Guide 2.

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 avail able 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 interchange-ability 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 sub-stance 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 [adapted from ISO/IEC Guide 2]

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 [adapted from ISO/IEC Guide 2]

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 [adapted from ISO/IEC Guide 2]

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.

3.1.9 Assurance of conformity

3.1.9.1
assurance of conformity
activity resulting in a statement giving confidence that a product, process or service fulfils specified requirements

NOTE For a product, the statement may be in the form of a document, a label or other equivalent means. It may also be printed in or applied on a communication, a catalogue, an invoice, a user instructions manual, etc. relating to the product.

3.1.9.2
supplier's declaration
procedure by which a supplier gives written assurance that a product, process or service conforms to specified requirements

NOTE In order to avoid any confusion, the expression "self-certification" should not be used.

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

The following terms have been incorporated or adapted from the ISO/IEC Directives, Part 2.

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 2382

The following terms have been incorporated or adapted from ISO/IEC 2382.

3.3.1 Fundamental terms [ISO/IEC 2382-01]

3.3.1.1
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).

3.3.1.2
interface
shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical interconnections, signal exchanges, and other characteristics, as appropriate

3.3.1.3
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.

3.3.1.4
data processing system
computer system
computing system
one or more computers, peripheral equipment, and software that perform data processing

3.3.1.5
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.

3.3.1.6
information processing system
one or more data processing systems and devices, such as office and communication equipment, that perform information processing

3.3.1.7
information system
information processing system, together with associated organizational resources such as human, technical, and financial resources, that provides and distributes information

3.3.1.8
to access
to obtain the use of a resource

3.3.1.9
application software
application program
software or a program that is specific to the solution of an application problem

3.3.1.10
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

3.3.1.11
compatibility
capability of a functional unit to meet the requirements of a specified interface without appreciable modification

NOTE This term differs from "compatibility" in ISO/IEC Guide 2 in that this term is specialized for use in information technology.

3.3.1.12
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

3.3.1.13
process 1
predetermined course of events defined by its purpose or by its effect, achieved under given conditions

3.3.1.14
process 2
«data processing» predetermined course of events that occur during the execution of all or part of a program

3.3.2 Organization of data [ISO/IEC 2382-04]

3.3.2.1
code, noun
collection of rules that maps the elements of a first set onto the elements of a second set

3.3.2.2
code value
code element
code (deprecated in this sense)
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 10646-1.

3.3.2.3
code set
code element set
code (deprecated in this sense)
result of applying a code to all elements of a coded set

EXAMPLES All the three-letter representations of airport names.

3.3.2.4
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.

3.3.2.5
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".

3.3.2.6
record
«organization of data» set of data elements treated as a unit

3.3.2.7
file
labeled set of records treated as a unit

3.3.3 Databases [ISO/IEC 2382-17]

3.3.3.1
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.

3.3.3.2
data model
description of the organization of data in the management information system of an enterprise

3.3.3.3
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

3.3.3.4
record
data object that is an instance of a record datatype

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

The following terms have been incorporated or adapted from ISO/IEC TR 10001-1.

3.4.1 Relationships among normative documents

3.4.1.1
base standard
approved standard used for creating derived standards [adapted from ISO/IEC TR 10000-1]

3.4.1.2
international standardized profile
ISP
internationally agreed-to, harmonized normative document which describes one or more profiles [adapted from ISO/IEC TR 10000-1]

3.4.1.3
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*.

3.4.1.4
IT system
set of IT resources providing services at one or more interfaces.

3.4.2 Conformance terminology

3.4.2.1
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

3.5 Terms and definitions from ISO/IEC 11404

The following terms have been incorporated or adapted from ISO/IEC 11404.

3.5.1 Fundamental concepts of datatypes

3.5.1.1
datatype
set of distinct values, characterized by properties of those values, and by operations on those values

3.5.1.2
value space
set of values for a given datatype

3.5.1.3
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().

3.5.2 Computational aspects of datatypes

3.5.2.1
regular value
element of a value space that is subject to a datatype's properties and characterizing operations

3.5.2.2
sentinel value
element of a value space that is not subject to a datatype's properties and characterizing operations

3.5.2.3
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

3.5.2.4
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 [adapted from ISO/IEC 11404]

3.5.3 Derived and generated datatypes

3.5.3.1
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

3.5.3.2
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

3.6 Terms and definitions from ISO/IEC 11179

The following terms have been incorporated or adapted from ISO/IEC 11179-1, ISO/IEC 11179-3, and ISO/IEC 11179-6.

3.6.1 General terms

3.6.1.1
metadata
data that defines and describes other data or processes

3.6.1.2
metadata registry
MDR
information system for registering metadata

3.6.2 Registry metamodel

3.6.2.1
administered item
registry item for which administrative information is recorded in an administration record

3.6.2.2
value domain
VD
set of permissible values

3.6.2.3
enumerated value domain
value domain that is specified by a list of all its permissible values

3.6.2.4
non-enumerated value domain
value domain that is specified by a description rather than a list of all permissible values

3.6.2.5
permissible value
ordered pair consisting of a value and its corresponding value meaning

3.6.2.6
value
data value

3.6.2.7
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.

3.6.3 Registration

3.6.3.1
registration
relationship between an administered item and the registration authority

3.6.3.2
registration authority
RA
organization responsible for maintaining a registry

3.6.3.3
registration status
designation of the status in the registration life-cycle of an administered item

3.7 Terms and definitions from ISO/IEC 20944

The following terms have been incorporated or adapted from ISO/IEC 20944-002 and ISO/IEC 20944-003.

3.7.1 Properties of implementations

3.7.1.1
implementation feature
artifact associated with an implementation

3.7.1.2
implementation value
quantifiable artifact associated with an implementation

3.7.1.3
implementation behavior
observable actions or appearance of an implementation

NOTE Implementation behavior may be specified by performance provisions.

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

3.7.2.1
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.

3.7.2.2
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

3.7.2.3
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

3.7.2.4
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

3.7.2.5
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

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.

NOTE An SPM sets a lower bound for implementations.

3.7.3 Deemed-to-satisfy provisions requiring implementation documentation

3.7.3.1
implementation-defined, adj
«technical specification» unspecified, yet each implementation documents how the choice among the available alternatives is made

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).

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).

3.7.3.2
implementation-defined value
«technical specification» unspecified value for which each implementation documents how the choice among the available alternatives is made

3.7.3.3
implementation-defined behavior
«technical specification» unspecified behavior for which each implementation documents how the choice among the available alternatives is made

EXAMPLE The exception handling associated with integer overflows; the exception handling with syntax errors on input data; the exception handling associated with file creation errors.

3.7.4 Deemed-to-satisfy provisions requiring implementation documentation

3.7.4.1
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.

3.7.4.2
undefined feature
«technical specification» implementation feature, within a content, for which a normative document imposes no requirements

3.7.4.3
undefined value
«technical specification» implementation value, within a content, for which a normative document imposes no requirements

3.7.4.4
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.

3.7.4.5
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 meaning. Thus, a feature may be: defined (but not unspecified), defined and unspecified, or undefined.

3.7.5 Implementation adaptation and specialization

3.7.5.1
locale
common characteristics and their properties for users based upon location

3.7.5.2
locale-specific, adj
changing, dependent upon locale

3.7.5.3
locale-specific feature
implementation-defined feature that changes, depending upon locale

3.7.5.4
locale-specific value
implementation-defined value that changes, depending upon locale

3.7.5.5
locale-specific behavior
implementation-defined behavior that changes, depending upon locale

3.7.5.6
user
human, his/her agent, a surrogate, or an entity that interacts with information processing systems

3.7.5.7
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.

3.7.5.8
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.

3.7.5.9
user-contextualized feature
implementation-defined behavior that changes, depending upon user context

3.7.5.10
user-contextualized value
implementation-defined value that changes, depending upon user context

3.7.5.11
user-contextualized behavior
implementation-defined behavior that changes, depending upon user context

3.7.6 Implementation documentation

3.7.6.1
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.

3.7.6.2
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.

3.7.7 Conformance kinds

3.7.7.1
conform
conforming
C
property, with respect to a normative document, that indicates the satisfaction of requirements

3.7.7.2
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.

3.7.7.3
comply
complying
compliance
property, with respect to a mandatory standard, that indicates the satisfaction of requirements

NOTE Thus, compliance concerns conformity to compulsory specifications, such as laws and regulations.

3.7.7.4
compliance
act of conforming to a mandatory standard

3.7.7.5
non-compliance
property, with respect to a mandatory standard, that indicates the lack of satisfaction of requirements

3.7.7.6
non-compliance
act of not conforming to a mandatory standard

3.7.7.7
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.

3.7.7.8
strict conformance
act of strictly conforming

3.7.7.9
nonconforming
not conform
not conforming
NC
property, with respect to a normative document, that indicates the lack of satisfaction of requirements

3.7.7.10
non-conformance
act of strictly conforming

3.7.7.11
merely conform
merely conforming
MC
property of conforming, but not strictly conforming

3.7.7.12
mere conformance
act of merely conforming

3.7.7.13
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).

3.7.8 Conformance framework

3.7.8.1
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.

3.7.8.2
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).

3.7.9 Data and information

3.7.9.1
data
instantiation of a relation between a concept and a sign whose concept includes a definition of an equality function

EXAMPLE 1 The sign "…" is associated with the concept of "image in monochrome pixels" and its data is a 2-dimensional array of bits with three of the bits (representing the dots in the ellipsis) are set to black and all others set to white.

EXAMPLE 2 The sign "…" is associated with the concept of "horizontal ellipsis" in ISO/IEC 10646-1 and its data is 2026 (in hexadecimal).

EXAMPLE 3 The sign "…" is associated with the concept of "the letter S" in Morse Code and its data is dot-dot-dot.

NOTE 1 The purpose of using the same sign in each of the examples is to show (1) how the concept influences the nature of the data (the relation between the concept and the sign), and (2) how different data can be created from the same sign.

NOTE 2 The first example is a simple set of pixels (sign) that creates data consisting of a set of bits (the 2-dimensional array). The second example is a sign that is related to a permissible value (each character's code value of ISO/IEC 10646-1 is a permissible value) that is part of a value domain (ISO/IEC 10646-1 character set) whose datatype is numeric. The third example is similar to the second example except that the datatype, fundamentally, is non-numeric. Of course, for all data that is computable in a digital data processing system, it is possible to map all data to numeric datatypes.

3.7.9.2
information
data in some context that gives it meaning

EXAMPLE 1 If X is a sequence of bits that represents an encrypted message (the data), there is no meaning (no information) until the decryption key is supplied. The decryption key is the context that gives meaning to the data (the sequence of bits).

EXAMPLE 2 20 is data, 20 becomes information when it is the "temperature in degrees Celsius of New York City at 2003-07-19 16:00 UTC".

3.7.9.3
MDIB
metadata interoperability and bindings

3.7.10 Conformance framework for data interchange

3.7.10.1
data interchange
concerning the representation, transmission, reception, storage, and retrieval of data

NOTE Data interchange may be though of a "information processing", excluding "data processing" (e.g., operations on data).

3.7.10.2
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.

3.7.10.3
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".

3.7.10.4
binding
result of applying or mapping one framework or specification to another

3.7.10.5
bound data instance
bound data
data instantiated from a datatype and rendered in a binding

3.7.10.6
data structure
bound data instance of an aggregate datatype

3.7.10.7
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.

3.7.10.8
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".

3.7.10.9
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".

3.7.10.10
data repository 1
functional unit that stores and retrieves data

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

3.7.10.11
data repository 2
collection of data

3.7.11 Data element obligation

3.7.11.1
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
```

3.7.11.2
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.

3.7.11.3
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.

3.7.11.4
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.

3.7.11.5
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).

3.7.12 Data element longevity

3.7.12.1
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.

3.7.12.2
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.

3.7.12.3
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.

3.7.12.4
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.

3.8 Terms and definitions particular to this document

The following terms are particular to this document.

3.8.1 Relationships among normative documents

3.8.1.1
base normative document
normative document used for creating derived normative documents

3.8.1.2
derived normative document
normative document that has one or more provisions in common with another normative document

3.8.1.3
base technical specification
normative document used for creating derived technical specifications

3.8.1.4
derived technical specification
technical specification that has one or more provisions in common with another normative document

3.8.1.5
derived standard
standard that has one or more provisions in common with another normative document

3.8.2 Extensions and their normative documents

3.8.2.1
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.

3.8.2.2
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

3.8.2.3
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

3.8.3 Relations among value spaces and value domains

3.8.3.1
extended value space
value space with addition elements, yet retains the same characterizing operations and datatype properties

3.8.3.2
isomorphic value domains
value domains such that there is a one-to-one and onto equivalence mapping of corresponding value meanings [adapted from ISO/IEC TR 20943-3]

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"** }.

3.8.3.3
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.

3.8.4 Relationships among value domains

3.8.4.1
base value domain
value domain used for creating derived normative documents

3.8.4.2
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.

3.8.4.3
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.

4 What is a normative document? a technical specification? a standard?

Definitions 3.1.3.1, 3.1.3.8, 3.1.3.2, define the terms normative document, technical specification, and standard. A technical specification and a standard are both examples of normative documents. A standard implies a consensus process which is approved by a recognized standards body. A technical specification includes "requirements to be fulfilled by a product, process or service", but does not imply consensus, e.g., a company's internal document might be a technical specification. Not all standards are technical specifications, e.g., a standard containing only terminology and no requirements is not a technical specification.

NOTE Throughout this document, the term "normative document" will be used as a general term unless the specialized terms "technical specification" or "standard" are required.

A normative document may be developed for a variety of reasons (see 3.1.2), such as fitness for purpose, compatibility, interchangeability, variety control.

A standard may be developed for particular geographic or political regions (see 3.1.3), such as: international standard, regional standard, national standard, provincial standard.

A prestandard (3.1.3.7) is standard developed for trial use.

Considering the international, multi-language, multi-cultural nature of standards, several kinds of equivalences among standards are possible (see 3.1.4), such as: harmonized standards (equivalent standards), unified standards, identical standards, comparable standards.

Normative documents are comprised of normative wording (see 3.1.5). The fundamental unit of normative wording is the provision, which may be a statement, instruction, recommendation, or requirement. Requirements may be exclusive requirements (mandatory requirement) or optional requirements. A deemed-to-satisfy provision is a requirement that permits several implementation alternatives (e.g., a minimum requirement).

5 What are data interchange standards and specifications?

Data interchange standards and specifications typically describe a data model (3.3.3.2), constraints (e.g., business rules), and data interoperability features (e.g., bindings to codings, APIs, and/or protocols).

Typically, a data interchange standard or specification is developed according to the needs of several stakeholders, such as users, vendors, organizations, and governments.

Data models may be expressed in terms of data elements (3.3.2.5). Based upon the stakeholders' needs for compatibility (3.1.2.2), interchangeability (3.1.2.3), and variety control (3.1.2.4), data elements are designated with various obligation attributes (see 3.7.11), such as mandatory data elements, optional data elements, and conditional data elements; and data elements are designated with various longevity attributes (see 3.7.12), such as reserved data elements, provisional data elements, and obsolete data elements. For metadata registries (3.6.1.2) that describe data elements, the registration status (3.6.3.3) may be describe the longevity attributes. As a practical measure, approaches to the long-term specification and maintenance are considered.[1]

Data elements are instances (3.7.10.3) of datatypes (3.5.1.1). A datatype consists of a value space (3.5.1.2), characterizing operations (3.5.1.3), and their properties. A value space consists of a set of values for a given datatype.

A value domain (3.6.2.2) is closely related to a value space: a value domain includes the meaning of each value. A value domain is a set of permissible values (3.6.2.5); a permissible value is the value (3.6.2.6) from the value space and its associated value meaning (3.6.2.7). Together the value and value meaning are known as a value-meaning pair.

Technical specifications should include procedures by means of which it may be determined whether the requirements given are fulfilled. Typically, these procedures are included in a conformance clause within the technical specification. The term conformity (3.1.8.1) concerns the fulfillment of the requirements. The term conformity assessment (3.1.8.2) concerns the measurement of this fulfillment of requirements.

Based upon the stakeholders' needs for compatibility, interchangeability and variety control, several kinds and aspects of conformance are specified. A technical specification may provide a wide range or a limited range of capabilities. Implementations may be characterized (see 3.7.1, 3.7.2, 3.7.3, 3.7.4, 3.7.5) by features, values, and behaviors that may be defined, undefined, unspecified, implementation-defined, local-dependent, and/or user-contextualized.

Implementations may describe their fulfillment of requirements via an implementation conformance statement (ICS) (3.4.2.1). The ICS may make claims of conformance, strict conformance, mere conformance, non-conformance and may provide additional implementation information (see 3.7.6, 3.7.7).

---

[1] The ISO/IEC 11179 (Metadata Registries) family of standards provide (1) a standard descriptive technique for data elements, value domains, and other data facets; (2) a common, interoperable registry-based approach for development and maintenance of the metadata. See "http://metadata-stds.org/11179".

During the development of a technical specification, several conformance roles (3.7.8.2) may be identified. For example, in a communications protocol standard, there may be two roles: client and server. Collectively, these conformance roles are known as a conformance paradigm (3.7.8.1). Rather than develop separate technical specifications that have cross-dependencies for normative wording (e.g., separate standards for the client side and the server side; both standards are certain to have cross-dependencies for wording), the A data interchange specification incorporates at least four conformance roles (see 3.7.10): data instance, data reader, data writer, and data repository.

Data interchange specifications map data models to some interoperability (3.3.1.10) interface (3.3.1.2) via bindings (3.7.10.4). A binding is the mapping of the data interchange specification to one or more coding bindings (e.g., XML, ASN.1), API bindings (e.g., C/C++/C#, Java, JavaScript), protocol bindings (e.g., ODBC, DCTP, SOAP), or other information infrastructure standards.[2]

Collectively, data interchange and interoperability are achieved when the following are addressed, characterized, and/or specified:

— The specification of the abstract data model.
— The decomposition of the data model into data elements, datatypes, value spaces, value domains, permissible values, and value meanings.
— The determination of obligation, longevity, and long-term maintenance of provisions, data elements, and value domains.
— The scoping and specification of defined, unspecified, and undefined features, values, and behaviors.
— The scoping and kinds of permitted implementation varieties, as described by implementation-defined, locale-specific, and user-contextualized features, values, and behaviors.
— The conformance procedures for determining and assessing conformity.
— The identification of a conformance framework, known as a conformance paradigm, and its conformance roles.
— The choice of interoperability bindings that affect data interchange.

NOTE The above list is not intended to be exhaustive or complete.

6 What issues arise when standards are adopted?

Users, vendors, and other stakeholders will take varying approaches toward the adoption of standards. Some stakeholders may desire "small" solutions, e.g., maximum interoperability, lower cost, tolerant of limited functionality, while other stakeholders may desire "large" solutions, e.g., full-functionality, tolerant of higher cost, tolerant of less interoperability. There is no *right* approach (small, large, or something in between), each approach satisfies a stakeholder's particular set of needs.

Because there are significant engineering considerations, typically, standards represent some balance and compromise. It might be impractical to choose the smallest set of features because, while everyone may achieve interoperability, the level of functionality is not enough to accomplish meaningful tasks. Also, it might be impractical to choose the largest set of features because, it may be too expensive to implement and it might be impossible to gain consensus on such a large set of features.

Thus, typically, standards represent some compromise of what can be accomplished and agreed upon *here and now*, with the expectation that further accomplishments are possible in the next revision of the standard. For practical reasons, stakeholders will experiment with new features (extensions, with respect to the

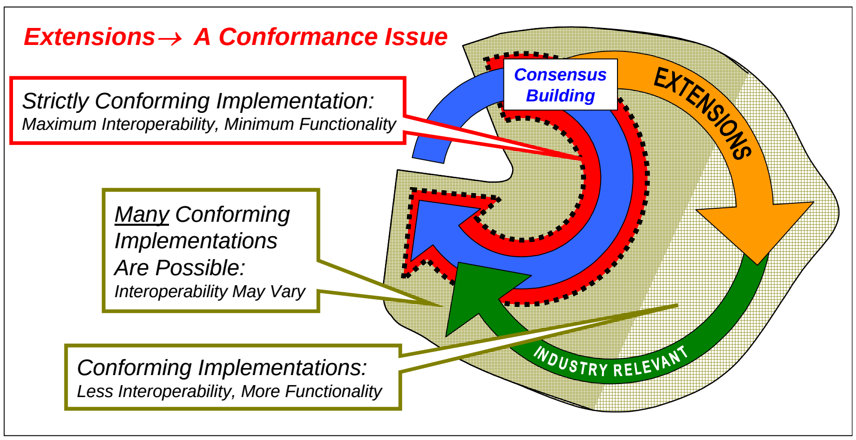
[2] For example, the ISO/IEC 20944 series of standards provide generic bindings of codings, APIs, and protocols. See "http://metadata-stds.org/20944".

standard) in preparation for the next revision of the standard. Some stakeholders may incorporate extensions to handle their private needs that are not specified (nor will they be specified) in the standard.

The general, preliminary findings of survey of IEEE LOM implementations (ISO/IEC JTC1 SC36 WG4 N0057) report some related findings:

— Fewer and better defined elements might be more effective than the range of choice and interpretive possibilities currently allowed by the LOM. This seems to be especially the case regarding educational elements, which are surprisingly underutilized for metadata that it ostensibly and primarily educational.
— The need for a smaller number of elements is further supported by the common identification of Dublin Core element equivalents in the application profiles surveyed.
— Clear and easily-supported means of working with local, customized vocabularies would also be very valuable — and that the means of retaining a minimum of interoperability between these variant vocabularies.
— It would be useful to ensure that structures are provided to accommodate complex but more conventional aspects of resource description. These would include multiple title versions, as well as multilingual descriptions and values.

The following diagram shows conformity issues overlaid on the standards development cycle.



7 Conformity, conformance

The use of conformance levels supports an industry's need to support a variety of implementations.

A conforming (3.7.7.1) implementation satisfies the requirements of a standard and may use extensions.

A strictly conforming (3.7.7.7) implementation satisfies the requirements of a standards and shall not use extensions.

A standard must balance the needs of interoperability with an industry's need for products that exceed the requirements of a standard. Because products with higher functionality (e.g., exceeding the requirements of a standard) also have lower interoperability (e.g., proprietary extensions), it is not enough to state that "an implementation conforms to the standard".

The distinction between "strictly conforming" and "conforming" implementations is necessary to address the simultaneous needs for interoperability and extensions. Standards promote interoperability. Extensions are motivated by needs of users, vendors, organizations, and industries (1) that are not directly specified by a standard, (2) that are specified and agreed to outside a standard, and (3) that may serve as trial usage for future editions of a standard.

An implementation that does not conform to a standard (either strictly conforming or merely conforming), is a non-conforming implementation.

8 Practical considerations that affect data modeling

The following discussion concerns the choice and the implications of particular data modeling provisions, such as obligation attributes and longevity attributes. This discussion includes the implications upon particular conformance roles in a data interchange conformance paradigm.

8.1 Obligation of data elements

There are four kinds of obligation attributes for data elements: mandatory, optional, conditional, and extended. The obligation attribute concerns the validity of the data structure.

8.1.1 Mandatory data elements

Mandatory data elements are always required for the data structure to be valid.

All data instances are required to include these elements. All data applications are required to support these elements.

An implementation that does not support or include one or more mandatory data elements is a non-conforming implementation.

8.1.2 Optional data elements

Optional data elements are permitted, but not required, for the data structure to be valid.

A data instance is permitted, but not required, to include these data elements. Because all data repositories and data readers are required to support all valid (strictly conforming) data instances, effectively, data repositories and data readers are required to support all optional data elements. This might be confusing because "optional" is not optional for data repositories and data readers — the obligation attribute "optional" applies to the validity of the data structure ("optional" is optional for instances of the aggregate datatype). A data writer is required to generate and produce the optional elements of each data instance that is generated and produced.

An implementation that includes or supports an optional data element, but includes or supports it in ways that are inconsistent with a normative document, is a non-conforming implementation. The attribute "optional" does not imply that the implementation has license to implement the data element in any way ("at the option of the implementor"); if the data element is implemented, its requirements are specified in a standard.

NOTE Thus, if optional data element **X** has the datatype `characterstring`, then either it doesn't exist or it exists as a `characterstring`; if **X** exists, but is not of the `characterstring` datatype, then the implementation is not conforming.

8.1.3 Conditional data elements

Conditional data elements are required, but their requirement is dependent upon certain conditions, as defined in a standard. Each conditional data element may individually have a set of conditions. If the conditions are met, the data element is required to be included for the data structure to be valid.

Thus, a data instance is required to include these elements if, individually, each condition is met. By the same reasoning as for optional data elements (above), all data repositories and data readers are required to support all conditional data elements. By the same reasoning above, a data writer is required to support all the conditional elements for each and every data instance generated and produced.

An implementation that includes or supports a conditional data element, but includes or supports it in ways that are inconsistent with a standard, is a non-conforming implementation.

8.1.4 Extended data elements

Extended data elements are not permitted within strictly conforming implementations.

Extended data elements are permitted within conforming implementations to the extent that the implementation individually supports each extended data element, i.e., (1) the implementation allows and uses specific extended data elements, (2) the data interchange participants allow and use specific data elements, and (3) other extended data elements are not used.

For conforming implementations that support extended elements, these elements individually may have their own obligation attributes, e.g., it is possible to have mandatory extended data elements, optional extended data elements, and conditional extended data elements. These obligation attributes determine the validity of the data structure in the context of extended data elements, e.g., an optional extended data element (1) permits but does not require the data element for the data structure to be valid, (2) for conforming implementations that support this extended data element.

NOTE Mandatory extended data elements can cause interoperability problems because a mandatory extended data element (1) requires the data element to exist for the data structure to be valid, (2) for conforming implementations that support this extended data element. In other words, (1) only implementations that support this extended data element are interoperable; and (2) no strictly conforming implementations will interoperate because extended features are required for interoperability.

There are no generic techniques or methods for both supporting extended data elements or extension features and supporting full semantic interoperability; there are only specific techniques and methods for supporting extended data elements (e.g., supported to the extent allowed, as above).

The use of extended data elements outside these circumstances (unsupported environments) causes undefined behavior, which might be:

— appropriate, e.g., ignoring an offending data element if it is an unimportant feature
— inappropriate, e.g., ignoring an offending data element if it is an important feature, such as a security classification
— innocuous, e.g., error messages
— disruptive, e.g., error messages
— predictable, e.g., a program aborting, exiting ungracefully, exiting unexpectedly, or "hanging" indefinitely
— unpredictable, e.g., a program aborting, exiting ungracefully, exiting unexpectedly, or "hanging" indefinitely

There is no correct generalized method for handling undefined behavior. Any particular method for handling undefined behavior can be desirable, undesirable, or both.

Some bindings "relax" the processing of unrecognized extended data elements. Normally, extended data elements create undefined behavior but certain bindings "relax" these requirements to implementation-defined behavior or even ignoring unrecognized extended data elements — both of these "relaxed" processing requirements (implementation-defined behavior; ignoring unrecognized or extended data elements) can be less disruptive.

Extended data elements are both an obligation (data modeling) feature and a conformance level feature (strictly conforming vs. conforming).

8.2 Longevity of data elements

The following longevity attributes indicate intentions for incorporation into past, present, or future editions of a standard.

Longevity attributes are independent of obligation attributes.

8.2.1 Obsolete data elements

Obsolete data elements are defined in the current edition of a standard and may be defined in prior editions. The "obsolete" feature indicates that the definition of the data element is intended to be removed from future editions of a standard.

Implementations should not use obsolete data elements. Implementations that do use obsolete data elements should plan accordingly for future editions of a standard.

An implementation's use of an obsolete data element does not imply that the implementation is non-conforming. Strictly conforming implementations and conforming implementations may still use obsolete data elements for this edition of a standard.

The "obsolete" feature is independent of the obligation attribute, so there might be obsolete mandatory data elements, obsolete optional data elements, obsolete conditional data elements, and obsolete extended data elements.

8.2.2 Reserved data elements

Reserved data elements are not defined in a standard. Data elements may be reserved because (1) they were defined in previous edition(s) of a standard, or (2) they will be defined in some future edition(s) of a standard.

A reserved data element is not permitted in a strictly conforming implementation.

A "reserved data element" might be used in a conforming implementation if (1) the reserved data element were defined, (2) it were defined as an extended data element, and (3) the extended data element were "supported" by implementations and data interchange participants (see below). In other words, a particular implementation extends or overrides (the non-definition of) the "reserved data element" by defining implementation extensions.

Although the "reserved" feature is independent of the obligation attribute, a reserved data element has no definition. Therefore, there are no reserved mandatory data elements, no reserved optional data elements, no reserved conditional data elements, and no reserved extended data elements because mandatory data elements, optional data elements, conditional data elements, and extended data elements all imply a definition of a data element, which conflicts with the undefined nature of "reserved".

Data elements that are defined, but are to be incorporated into future editions of a standard, are extended data elements (i.e., they are not reserved data elements). As these extended data elements become incorporated into a future edition, they will become mandatory data elements, optional data elements, or conditional data elements. Additionally, provisional data elements may be used in transition to a new edition of a standard.

Typically, extended data elements are defined in a specification outside the standard.

Extended data elements are not required for this edition of a standard, i.e., (1) extended data elements are prohibited for strictly conforming systems; (2) extended data elements are not required for conforming systems; and (3) extended data elements, if defined, are not in a standard.

Some bindings "relax" the processing of unrecognized data elements, such as reserved data elements. Normally, reserved data elements create undefined behavior but certain bindings "relax" these requirements to implementation-defined behavior or even ignoring unrecognized or reserved data elements — both of these "relaxed" processing requirements (implementation-defined behavior; ignoring unrecognized or reserved data elements) can be less disruptive.

Conforming implementations may use extended data elements to the extent permitted by the implementation and data interchange participants. See "Extended Data Elements" (above) for further details.

Reserving a data element so that it cannot be overridden by extended data elements is achieved by defining an optional data element with ISO/IEC 11404 datatype **`void`**.

8.3 Recursive and contextual nature of obligation and longevity

An obligation attribute or a longevity attribute of an aggregate data element applies to the aggregate itself, but only indirectly to its components. In the context of the existence of an aggregate and its components, each component individually has its own obligation and longevity attributes (among other attributes). This determination of context and obligation/longevity attributes is applied recursively for all aggregate data elements.

EXAMPLE A data element **X** is optional, and **X** has two subelements: **Y** is mandatory and **Z** is optional. Letting the notation **P.Q** represent the subelement **Q** of **P**, then

— if **X** does not exist, then **X.Y** and **X.Z** cannot exist; stated differently, if **X.Y** or **X.Z** exists, then **X** exists
— if **X** exists, then **X.Y** is required to exist for all conforming implementations
— if **X** exists, then **X.Z** is permitted to exist for all conforming implementations
— if **X** exists and **X.Y** does not exist, then the implementation is non-conforming

Thus, **Y** only becomes mandatory if **X** exists.

9 What is an extension?

An extension (3.8.2.1) is a derived normative document (3.8.1.2) that includes additional provisions with respect to a base normative document (3.8.1.1). Extensions may produce more functionality or less functionality. Based on the definition of "extension", there are no implications of any kind of interoperability or compatibility between implementations of the base normative document and implementations of the derived normative document.

NOTE It is a common misunderstanding that an "extension" only provides additional capability. The perception of "additional" is dependent upon the stakeholder's perspective, e.g., one stakeholder's additional capability may be another stakeholder's diminished capability.

Simply creating extensions (in a formal sense) still leaves uncertainty for interoperability and compatibility, so further provisions are necessary to create interoperability and compatibility.

9.1 Extensions for value spaces

An extend value space (3.8.3.1) for a datatype implies compatibility in characterizing operations and properties. However, values value space may be tied to value meanings of a value domain. The interoperability and compatibility of the newly derived value domain (3.8.4.2) is indeterminate because the similarity of change in value meanings (for the extended value space) has not been specified.

9.2 Extensions for value domains

Extending a value domain is not as simple as extending a value space. Isomorphic value domains (3.8.3.2) are equivalent in meaning, but may have different values (e.g., codes). The creating of isomorphic value domains may be useful for representing code element mappings.

An overlapping value domain (3.8.3.3) shares some permissible values with another value domain (for an example of three overlapping value domains, see the diagram in 10.3 on the value domain for "colors").

An extended value domain (3.8.4.3) is a superset of the base value domain requires its permissible values to retain their meanings with respect to the base value domain. For value domains that represent a (mathematical) partition, i.e., a classification that is comprehensive and not overlapping, it is difficult to extend the value domain without changing the meaning (changing the meaning introduces interoperability and compatibility problems).

9.3 Extensions for data elements

With respect to a base normative document for data interchange, it is possible to create a subsets (3.8.2.2) and supersets (3.8.2.3) for normative documents, technical specifications, and standards (e.g., a "subset technical specification" or a "superset standard"). Both subsets and supersets are considered extensions (i.e., there are additional provisions). A subset data interchange specification implies a subset of conforming data instances; a superset data interchange specification implies a superset of conforming data instances.

Thus, the "subset" and "superset" notions make assertions about interoperability and compatibility between the base normative document and the derived normative document.

10 What issues arise when using extensions?

The following illustrations concern extensions for data elements and extensions for value domains.

10.1 Illustration: A notion of "core" data elements

Some vendors, users, organizations, industries, etc., decide they need a "core set" of elements with respect to some standard. The term "core set", typically, means (1) a subset of the original data elements are only required, (2) stronger requirements are made for the "core set" of data elements, (3) optional extended data elements, or (4) some combination. Assuming the following record is defined in a standard:

```
STD: record  // definition specified in the standard
(
    A: integer,  // optional data element
    B: integer,  // optional data element
    C: integer,  // optional data element
)
```

The "core set" might only include data elements **B** and **C**, but these data elements would become mandatory.

```
CORE: record  // "core set" definition
(
    A: integer,  // mandatory data element
    B: integer,  // mandatory data element
)
```

Applications that merely conform to the **CORE** specification will not be able to store and process records that conform to the **STD** standard. Example 1: An **STD** record that includes data element **C** will lose that element when stored in a **CORE** repository. Example 2: A conforming **STD** record that does not include the optional data element **B** will be rejected by a **CORE** repository because **B** is mandatory for **CORE**.

Although it may appear that creating a "core set" optimizes the data model, this notion of a "core set" creates significant interoperability problems, which can have a negative impact on industry adoption of a standard.

The following are several approaches and techniques for handling many of the competing needs of extensions and extended data elements.

In other words, if Organization is creating a "core set", then Organization needs to carefully craft its words to make sure its extended records include those records that conform to the standard records, e.g., do all standard records also conform to Organization's "core set" and will they remain unstripped by the Organization's data repositories? This area of compatibility and interoperability is often confused.

In these three examples, the first example is a reasonable approach for the Organization to extending the standard record, i.e., just adding more elements.

The second example, which addresses "core sets", might be reasonable a reasonable approach towards extending standard records, but has this example still has some hazards that might or might not be worked around.

The third example of extending standard records, which also addresses "core sets", has some significant hazards and would be a "surprise" to most implementers of the standard (vendor, organizations, etc.), i.e., information is stripped from its original source. This approach towards "core sets" causes problems.

Organizations considering extensions and "core sets" should migrate towards the first two examples and should migrate away from the third example.

Another way of inquiring about these compatibility issues for extensions is to ask two questions (need YES answers to both questions):

- Will all records that conform to the standard record be acceptable to (conform to) the specification that describes the Organization extensions?
- For all records that conform to a standard, when stored in standard repositories, will these records contain at least the same elements from the original record with the original values of the said elements? In other words, for an standard record **P** that is stored in an Organization-specific repository as record **Q**, is it true, applied recursively, that for each element **P** there exists the same element in **Q** with the same value?

The term "core set" should be avoided because it can be confusing and it may mean the wrong (undesirable) thing.

Compatibility issues should be carefully analyzed to make sure that the creators of conforming organization-specific records do not have their information lost or changed.

Creating variants like the third example, **ORG_3** (below), will cause the splintering of standards.

Example #1

For example, say the standard records are specified as follows:

```
STD: record  // definition specified in the standard
(
```

```
        A: integer,  // optional data element
        B: integer,  // optional data element
        C: integer,  // optional data element
    )
```

Now let's say that a vendor, organization, user, etc., wants to create their own specialized version of this standard record. This might make sense for a variety of reasons. For vendors, extensions might create additional functionality that might make their product more attractive. For organizations, extensions might support organization-specific features that are not part of the standard. The following might be an organizational extension:

```
    ORG_1: record  // organizational variant #1
    (
        A: integer,  // optional data element
        B: integer,  // optional data element
        C: integer,  // optional data element
        D: integer,  // optional, organizational extension
    )
```

In this organizational extension, standard records might have an additional element D. The following are samples of conforming records.

```
    // SC = strictly conforming
    // C = conforming
    // NC = non-conforming
    { A }               // SC/C to STD, SC/C to ORG_1
    { B }               // SC/C to STD, SC/C to ORG_1
    { A, B, C }         // SC/C to STD, SC/C to ORG_1
    { A, D }            // C to STD, SC/C to ORG_1
    { A, B, C, D }      // C to STD, SC/C to ORG_1
    { A, B, C, D, E }   // C to STD, C to ORG_1
```

Note that the last three examples contain extensions to the standard record, so they are only "conforming" (not "strictly conforming"), and the last example has the extension **E** (e.g., a vendor extension) that is both outside the standard records and **ORG_1**'s version of the record. The example above is a typical extension that causes no problems.

Conclusion: This approach is a reasonable organizational extension.

Example #2

Another possible approach is for organizations (or vendors, etc.) to make certain fields "mandatory" via some notion of a "core set". The main purpose of this might be to recognize that certain repositories have certain useful "keys", e.g., the mandatory fields become database keys which may optimize searches.

Note that the need for certain fields may be a data typing issue, a data quality issue, or both. This discussion only concerns data typing issues and does not discuss data quality issues, which are outside the scope of data extensions. Trying to solve a technical issue (data typing) with a management issue (data quality), or vice versa, is likely to fail. These are separate issues that should be addressed separately.

In this case, the data structure that defines the organizational specification might look like:

```
    ORG_2: record  // organizational variant #2
    (
        A: integer,  // mandatory data element
        B: integer,  // mandatory data element
        C: integer,  // optional data element
        D: integer,  // optional, organizational extension
```

```
)
```

Note that the **ORG_2** specification invalidates certain records that conform to standard:

```
// SC = strictly conforming
// C = conforming
// NC = non-conforming
{ A }               // SC/C to STD, NC to ORG_2
{ B }               // SC/C to STD, NC to ORG_2
{ A, D }            // C to STD, NC to ORG_2
```

In other words, those users and vendors that created standard records (and, created them to strictly conform to the standard) now find their records rejected by **ORG_2** because they are non-conforming to **ORG_2** (missing some mandatory elements). A typical workaround to this scenario is that standard records that arrive at this organization are mechanically transformed by supplying **nil** or dummy fields:

```
// SC = strictly conforming
// C = conforming
// NC = non-conforming
{ A } ==>
  { A, B=nil }      // SC/C to STD, SC/C to ORG_2
{ B } ==>
  { A=nil, B }      // SC/C to STD, SC/C ORG_2
```

One area that is missing from many data model standards is the notion of "equivalence" between records, i.e., How does one determine that record P and record Q are the same? For example, are the following two records the same?

```
{ J }
{ J, K=nil }
```

These two records might or might not be considered the same.

Conclusion: ORG_2 might cause some problems because existing records that conform to the standard might be rejected by **ORG_2**. A simple workaround is to create nil/dummy fields, but (1) it is ambiguous as to whether or not these new records are equivalent, and (2) if these records are not equivalent, there might be semantic, legal, and interoperability concerns.

Example #3

Another third possible approach is for organizations (or vendors, etc.) keep a "core set" of important elements and eliminate "unnecessary" or "unused" elements.

```
ORG_3: record  // organizational variant #3
(
    A: integer,  // mandatory data element
    B: integer,  // mandatory data element
    // C is missing because it is not considered "core"
    D: integer,  // optional, organizational extension
)
```

This notion of a "core set" is one that trims down the standard record only to its so-called "most important" elements. This approach does conform to original standards. In the following examples:

```
// SC = strictly conforming
// C = conforming
// NC = non-conforming
{ A }               // SC/C to STD, NC to ORG_3
```



```
{ B }                 // SC/C to STD, NC to ORG_3
{ A, B, C }           // SC/C to STD, **C** to ORG_3
{ A, D }              // C to STD, NC to ORG_3
{ A, B, C, D }        // C to STD, **C** to ORG_3
```

The surprise here is that **{ A, B, C }** strictly conforms to the standard, but only conforms to the **ORG_3** (a similar problem exists for **{ A, B, C, D }**). Stated differently, all systems that conform to the standard must (shall) be able to store **{ A, B, C }**. Systems that conform to **ORG_3** specification are not required to accept the record **{ A, B, C }** or may just strip the record of element **C**.

This is a problem for content developers, publishers, and learning management system developers because (1) they created the data according to the standard, and (2) some of the data was discarded or changed.

Conclusion: Just because an element is "optional" doesn't mean that the storage/retrieval of element is optional. The **ORG_3** approach will definitely cause problems for content and for software that are dependent on these lost/changed fields. This issue arises in:

— conformance testing: may pass **ORG_3** conformance tests, but will fail standard conformance tests
— procurement: need to watch for vendors taking shortcuts
— interoperability: loss/change of information can cause operational/legal problems

The **ORG_3** approach is incompatible with the standard record.

10.2 Illustration: Value domains and their extensions

Value domains consist of permissible values. Each permissible value has a corresponding meaning. Value domains may be enumerated (all their permissible values and meanings are itemized) or non-enumerated (their values and meanings are defined by rules).

Unlike data elements, which are easier to extend, value domains are more complex by their nature. If one considers a "value domain" as a classification of data (i.e., observations or instrumentation that produces "data" as one of the classes), then changing the classification, such as adding a new permissible value, changes the meaning of the data. The following illustration (excerpted from SC36/N0425) demonstrates this point by showing how the permissible value "single" has a different meaning, dependent upon the value domain:

> [Excerpt[3] from SC36/N0425, *"Recommendation on the Internationalization of Data Elements, Value Domains, and Permissible Values"*]
>
> Illustration: The permissible values **single** and **married** within a value domain for marital status.
>
> The permissible value **single** may have different meanings, depending upon the definition of the value domain, e.g., value domain #1 **{ single, married }** and value domain #2 **{ single, married, divorced, widowed }** have different meanings for the permissible value **single**. In other words, it is impossible to determine the meaning of **single** unless one knows which value domain **single** is from.

Thus, if one takes a "standard" value domain that contains **{ single, married }** and extends it with two additional permissible values, **{ single, married, divorced, widowed }**, then the meanings of the original permissible values have changed for the datatype <u>*and for the data itself*</u>. Thus, there are interoperability

[3] Available at "http://jtc1sc36.org/doc/36N0425.pdf".

problems because (1) there is disagreement on the meaning of data, and (2) the meaning of the data has changed.

However, the need to extend value domains is important when adopting international standards and applying them to specific national, cultural, or organizational needs.

An important characteristic of classifications is their notion of "comprehensiveness". A classification is considered "comprehensive" when its classes cover all possibilities (also known mathematically as a "partition"). For example, in a particular society the classification of **{ single, married, divorced, widowed }** is intended to cover all possible martial statuses, i.e., for every person, their marital status can be mapped into one of those four possibilities. As described above, extending a value domain of a comprehensive classification causes the meaning to change of the permissible values.

Classifications are considered "non-comprehensive" when their classes do not cover all possibilities. For example, a value domain concerning "breakfast" contains **{ eggs, cereal, fruit }**, i.e., a person may order eggs from the "standard breakfast". There may be cultural variants that extend this value domain, such as:

```
New York Extension: { eggs, cereal, fruit, bagel }
Copenhagen Extension: { eggs, cereal, fruit, herring }
Tokyo Extension: { eggs, cereal, fruit, miso soup }
```

Note that none of the above cultural extensions cause any conflicts. However, the following cultural extension

```
Honolulu Extension: { eggs, cereal, fruit, pineapple }
```

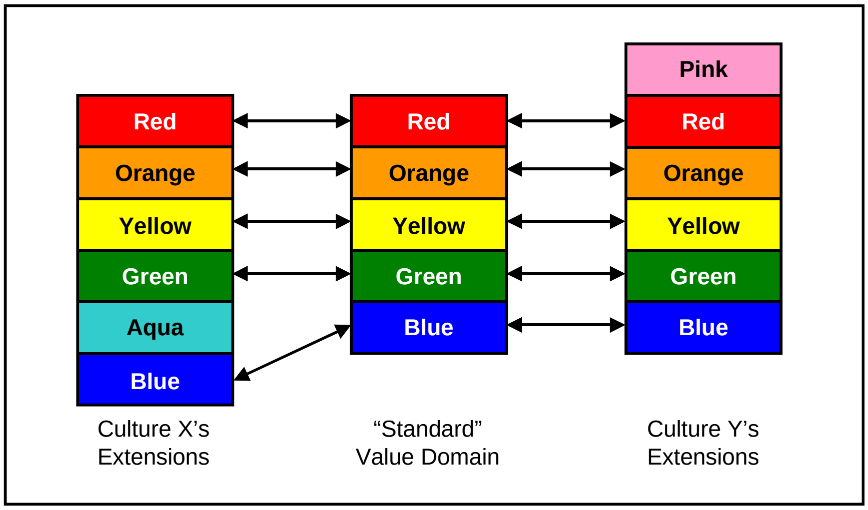
does cause problems because **fruit** and **pineapple** overlap from a classification perspective, i.e., an object may be classified as both a **fruit** and a **pineapple**. A more insidious problem is the following cultural extension

```
Helsinki Extension: { eggs, cereal, fruit, porridge }
```

because for Finland, **cereal** and **porridge** are considered separate and non-overlapping, but for the United States, **cereal** and **porridge** are considered overlapping (i.e., **porridge** is a kind of **cereal**). These kind of problems are not limited to extensions, but may be embedded in base standards due to poor internationalization or harmonization efforts.[4]

10.3 Illustration: Harmonized culture-specific value domains

[4] These kind of internationalization issues were reported in the review of the IEEE LOM draft 6.4. See SC36/N0276 at "http://jtc1sc36.org/doc/36N0276.pdf".



One approach to international harmonization is creating several value domains that are related.[5] The diagram above shows how various cultures' (or nations') extensions may be related to a "standard" value domain, i.e., the meaning of "blue" in culture X is related to the meaning of "blue" in the "standard", and is then related to the meaning of "blue" in culture Y. With this approach, the complexity of interoperability of value domain extensions is reduced from complexity N2 to N.

In conclusion, extensions to value domains requires careful consideration with respect to (1) comprehensiveness of the value domain, and (2) overlap of individual permissible values and their meanings.

10.4 Illustration: A multi-phase approach to processing data

Implementations that separate data readers into the translation phases of consumption and interpretation and data writers into the translation phases of generation and production may allow certain bindings to better handle extensions.

In terms of an implementation's behavior, a consistent framework can specify data generation semantics and data interpretation semantics, yet certain interoperability bindings allow the "behavior" to be relaxed by producing less undefined behavior.

EXAMPLE 1 XML coding bindings can ignore unknown tags in data consumption.

EXAMPLE 2 Other bindings can handle (ignore, diagnose, etc.) extension prefixes, e.g., identifiers that begin with "X-", "x-", or "__" (double underscores).

11 What is a profile? What is a derived standard?

A "profile" (3.4.1.3) 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, or ISPs necessary to accomplish a particular function".

Profiles reference other standards (see 3.1.7). References may be dated (3.1.7.2) or undated (3.1.7.3).

---

[5] ISO/IEC 11179-3 (Metadata Registries, Registry Metamodel) provides a standard technique for expressing relationships among several value domains.

11.1 General principles of a profile

The general principles of a profile are specified in ISO/IEC TR 10000-1, 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 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.
>
> Thus conformance to a profile implies by definition conformance to the set of base standards which it references. However, conformance to that set of base standards does not necessarily imply conformance to the profile.

While the last paragraph above summarizes one important aspect of interoperability and compatibility (i.e., conformance to the profile implies conformance to the base standard), from the perspective of the developer of profile, a more important interoperability and compatibility issue is item b above: <u>*[a profile] shall not specify any requirements that would contradict or cause non-conformance to the base standards to which it refers*</u>. This requirement has a profound effect upon profiles of data interchange standards because profiles inherit certain <u>*implicit*</u> requirements from base standards (see Clause 12, below).

11.2 Main elements of a profile definition

The main elements of a profile definition are specified in ISO/IEC TR 10000-1, subclause 6.3.2:

> **6.3.2 Main elements of a profile definition**
>
> The definition of a profile shall comprise the following elements:
>
> a) a concise definition of the scope of the function for which the profile is defined and the user requirements which it will satisfy, which is capable of being used as an Executive Summary of the profile;
>
> b) an illustration of the scenario within which the profile is applicable, giving, where possible, a diagrammatic representation of the IT systems, applications and interfaces which are relevant;
>
> c) normative reference to a single set of base standards or ISPs, including precise identification of the actual texts of the base standards or ISPs being used; also identification of any approved amendments and technical corrigenda (errata), conformance to which is identified as potentially having an impact on achieving interoperability or portability using the profile;

d) specifications of the application of each referenced base standard or ISP, stating the choice of classes or conforming subsets, and the selection of options, ranges of parameter values, etc, and reference to registered objects;

e) a statement defining the requirements to be observed by IT systems claiming conformance to the profile, including any remaining permitted options of the referenced base standards or ISPs, which thus become options of the profile;

f) if relevant, a reference to the specification of conformance tests for the profile;

g) informative reference to any amendments or technical corrigenda to the base standards referenced in the profile, which have been determined to be not applicable to the profile, and to any other relevant source documents

11.3 Derived standards

A derived standard (3.8.1.5) is a normative document that has provisions in common with a base standard. In comparison to profiles, a derived standard makes no requirements concerned the relationship between conformance to the base standard and conformance to the derived standard (or vice versa). This an implementation that conforms to the derived standard is not required to conform to the base standard (and vice versa).

11.4 Copy/paste vs. incorporation via normative reference

From the perspective of standards interpretation and the meaning of a profile, there is no difference between copying and pasting normative wording into the profile vs. incorporating provisions via normative reference (i.e., reference to Clause, subclause, etc.). This non-distinction applies to both profiles and derived standards.

The decision of copy/paste vs. incorporation via normative reference is affected by standards maintenance and the availability of base standards. Generally, incorporation via normative reference is preferred because (1) it minimizes the editing and review work of the profile or derived standard, (2) the maintenance is (largely) the responsibility of the committee that developed the base standard, (3) technical corrigenda and amendments of the base standard may be incorporated into the profile or derived standard without modifying (and balloting) the new document. In some cases copy/paste may be preferable, such as (1) when, due to the structure of the base standard, the normative referencing is complex or impractical, (2) when the normative wording in the base document may be unavailable, such as references to specifications other than international standards (see document JTC1/N4046 and JTC1/N4047).

12 What issues arise when using profiles and derived standards?

The following is a discussion of issues that may arise when using profiles and/or derived standards.

12.1 Profiles vs. derived standards for subset standards

As an illustration, consider a base standard for data interchange that specifies a record containing 5 optional data elements: **name**, **address**, **city**, **state-province**, **post-code**. A profile of this standard only includes the last four elements, still all optional data elements: **address**, **city**, **state-province**, **post-code**. While the profile might have obvious consequences concerning conforming data instances and data writers (for both, a data instance may contain between 0 and 4 of the optional data elements), the consequences are less obvious for data readers and data repositories because of implicit requirements inherited from the base standard — inherited because of a profile's requirements, as specified in ISO/IEC TR 10001-1, subclause

6.3.1, item b: *[a profile] shall not specify any requirements that would contradict or cause non-conformance to the base standards to which it refers*.

In the base standard, there are there are four conformance roles, each with their own set of requirements. In the base standard, the data instance conformance role (and, similarly, the data writer conformance role) has the following implied requirements:

— **name** is permitted, but not required, for a data instance
— **address** is permitted, but not required, for a data instance
— **city** is permitted, but not required, for a data instance
— **state-province** is permitted, but not required, for a data instance
— **post-code** is permitted, but not required, for a data instance

i.e., there are no requirements. The data repository conformance role (and, similarly, the data reader role) has the following implied requirements:

— **name** is required for a data repository
— **address** is required for a data repository
— **city** is required for a data repository
— **state-province** is required for a data repository
— **post-code** is required for a data repository

i.e., ***5*** requirements. As described above, all optional data elements shall be implemented (i.e., are mandatory) in data repositories.

A conforming implementation of a data instance of the above profile makes no requirements on the data element **name** because the base standard makes no requirements for a data instance (i.e., "... is permitted, but not required, ..." makes no requirements), thus no requirements are implicitly inherited into the profile and **name** is not required in a data instance of the profile standard.

A conforming implementation of a data repository of the above profile makes ***4*** requirements (i.e., requiring the data elements **address**, **city**, **state-province**, **post-code**) and then the profile itself inherits requirements from the base standard, i.e., making ***5*** requirements (i.e., requiring all ***5*** data elements, including **name**). The reason why **name** cannot be excluded (and, thus, must be included in a conforming data repository *of the profile*) is that requirements cannot be eliminated in profiles, is that profiles "shall not specify any requirements that would contradict ... the base standards to which it refers" and the elimination of a requirement would contradict the base standard.

So a conforming data repository for the profile of ***4*** optional data elements requires the ***5*** data elements of the base standard (not the ***4*** data elements of the profile).

In conclusion, all the data elements of the base standard are required all profiles' conforming data repositories. This outcome (always requiring all the data elements in a conforming data repository, regardless of how small the profile is) may be desirable (for those who want higher interoperability and compatibility) and may be undesirable (for those who perceive the additional data elements as significant added cost). The full set of data elements is required in data repository because *the standard is called a profile* (which has certain conformance requirements, as specified by ISO/IEC TR 10000-1, subclause 6.3.1, item b. If the standard were merely a derived standard, there are no additional implicit requirements, so it is possible to define a new record that only contains 4 optional data elements (**address**, **city**, **state-province**, **post-code**) and for a data repository that conforms to the derived standard to only require the same 4 data elements. In other words, if the goal is to reduce the number of data elements required in a data repository, then *create a derived standard, not a profile*.

12.2 Application and use of extensions in "profiles"

There are widely used phrase “Metadata Application Profile” describing metadata elements selected from one or more metadata schemas and combined in a compound schema. These profiles tend to define some elements as mandatory (“core” elements discussed in previous chapter) and give some in-depth (relevant to the community that is using the profile) information about elements and their value spaces. Different communities of practice have adopted this profile approach in their metadata needs: e.g. CanCore[6], CELEBRATE[7], CLEO[8], Curriculum Online[9], and The Le@rning Federation[10]. Various requirements of their metadata needs have forced them also to introduce extension data elements and extensions to the value spaces of the elements of the standards they are using. These extensions reflect specific community’s point of view and do not pose interoperability issues inside the community that is using them but although the documentation of the "application profiles" are usually quite extensive, interoperability between different communities is very laborious and manual process. However, because in standards it is impossible to support arbitrary extensions in a generic way that provides meaning to the extensions, "application profiles" provide important means of profiling standards for the needs of particular communities. The development and implementation of whole "application profiles" are out of scope of this document, but this chapter explains several approaches and techniques for handling many of the competing needs of extensions and extended data elements.

12.3 Supporting several implementation models

A single type of conformance does not address all technical, business, and interoperability needs. The notions of data instances, data repositories, data readers, and data writers allow implementers to choose the type of conformance claim they desire.

EXAMPLE 1 Tools that export data will only want to claim conformance as data writers and not be burdened with the requirements of a data repository — the vendor doesn't want all that functionality and the consumer doesn't need it.

EXAMPLE 2 A vendor of a database system might want to claim conformance as a data repository and distinguish their product from other products — the vendor does want all that functionality and the consumer wants it, too.

Extensions and extended data elements raise conformance issues.

A standard must balance the needs of interoperability with an industry's need for products that exceed the requirements of a standard. Because products with higher functionality (e.g., exceeding the requirements of a standard) also have lower interoperability (e.g., proprietary extensions), it is not enough to state that "an implementation conforms to the standard".

A strictly conforming implementation has higher interoperability (with respect to the standard), but lower functionality. An implementation that is merely conforming (but not strictly conforming) may have higher functionality but lower interoperability with respect to this Guideline.

Because implementations are labeled "strictly conforming" or merely "conforming", the consumer and the marketplace can make appropriate choices among varying qualities of implementation.

---

[6] http://www.cancore.ca

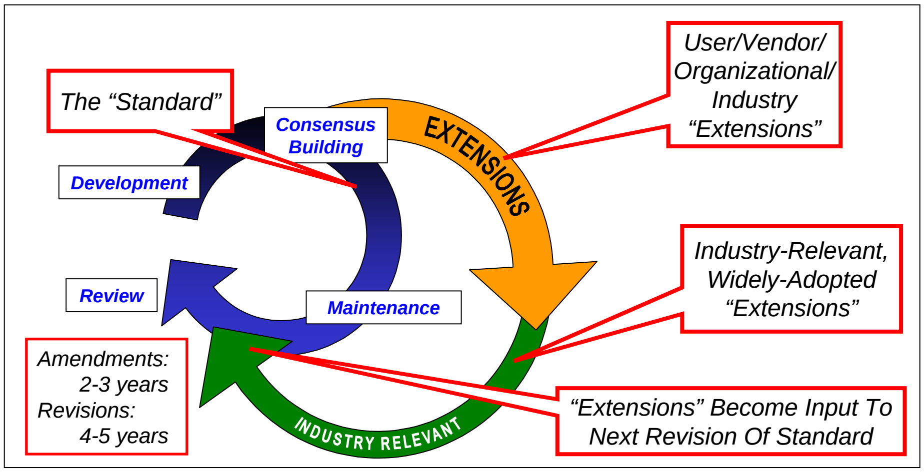
[7] http://celebrate.eun.org

[8] http://www.cleolab.org

[9] http://www.curriculumonline.gov.uk

[10] http://www.thelearningfederation.edu.au

12.4 Incorporating extensions via the standards development cycle



Extensions and extended data elements are techniques that industries use to "try out" new features and "let the market decide". When certain extensions become industry relevant, they may be incorporated into an amendment (typically, 2-3 years after a standard is approved) or into a revision (typically, 4-5 years after a standard is approved).

It is easier to build consensus around features that have been widely adopted than those that have not yet been widely adopted. This incremental approach may help "grow" a standard's data model over time as an industry increasingly adopts improvements into the data model.

For value domains that represent (mathematical) partitions, the techniques for extending the value domain are different than extending data elements in an aggregate data type. For precise classification work, a common approach is to use a (so-called) "4-D model" rather than a "3-D model" of data.[11] Typically, when using a value domain for classification, the analytical results would reference the particular *version* (or edition) of the value domain. As an illustration, if one performs a survey and asks the question "What country are you from?", *the answer will depend upon when the question was asked*, e.g., a classification in 1980 would have distinguished between the classes **West Germany** and **East Germany**, but a classification in 2000 (or 1900) would only include the class **Germany**. The complexity of processing of a time-dependent value domain will vary depending upon application. A simple scenario might be storing birth records: simply recording the birth date, the version of the value domain (list of countries), and the country of birth. A complex application might be one in which the results of a classification performed decades ago, is projected onto (say) the present moment: how is "legacy" data transformed and made comparable to present data?[12]

---

[11] The notation of "4-D" vs. "3-D" does not actually mean 4 dimensions vs. 3 dimensions. This notion is commonly used to convey the distinction between time-dependent and time-independent models.

[12] Time-sensitive classification causes significant interoperability and compatibility problems. One of the comments on IEEE LOM Draft 6.4 (see SC36/N0276) was that it created a "guild" (a list of professions — a classification) that were particularly relevant circa 1999, but might be significantly different ten years later, i.e., a very time-sensitive classification. Aside from the classification concerns, the comment also noted that this kind of "guilding" might itself create long-term artificial professional designations that are intended to coincide with the developers' roles, as recorded in the metadata.

13 Comparison of extensions vs. profiles

Standards developers and stakeholders may have an interest in developing extensions, profiles, or both. It is important to clarify the difference between an "extension" and a "profile": the former concerns in-scope or out-of-scope features with respect to a base technical specification, while the latter concerns the conformance relationship between two technical specifications.

The survey of metadata implementations (ISO/IEC JTC1 SC36 WG4 N0057) reports on the progress and preliminary findings of an ongoing international survey of Learning Object Metadata (LOM) implementations. Many of the records surveyed have been created through reference to an “application profile”.[13] Application profiles draw from both the LOM and Dublin Core and are defined primarily in terms of data element subsets.[14] Considering issues presented in this document these “application profiles” are not “profiles” as presented here because they are not concerned about conformance, but they are “extensions” of the base standard.[15] However, the term "application profile" is used extensively in the metadata community and there is therefore a challenge to harmonize terminology."

14 Conclusions and recommendations

Extensions may be desired by many kinds of stakeholders (users, vendors, organizations, cultures, nations, regions), but (1) they can have a profound effect on interoperability, and (2) they require careful consideration.

Extensions may be described by descriptive techniques, such as 11179-3 and RDF.

Conformance is not determined from a single perspective, but from many perspectives (the conformance paradigm), such as data readers, data writers, data repositories, and data itself.

Conformance has different meanings, dependent upon the perspective, e.g., while a data element may be optional within a data instance, that data element's understanding and processing is mandatory within a data repository.

For interoperability and commercial reasons, it is important to distinguish among at two levels of conformance, such as strictly conforming (no extensions), and conforming (extensions are possible).

The relationship between implementations that conform to base standards vs. implementations that conform to derived standards (or profiles) needs to be carefully considered so that interoperability, compatibility, and conformance are understood and agreed upon.

---

[13] In the survey, the term "application profile" is used inconsistently and, in some cases, imprecisely. Roughly, "application profile" was used to mean one or more of the following: "profile" (3.4.1.3), "derived standard" (3.8.1.5), "extension" (3.8.2.1), "subset standard" (3.8.2.2), "superset standard" (3.8.2.3).

[14] These "data element subsets" might be intended to be a "subset standard" (3.8.2.2), but further specification is required.

[15] An important result of the survey was the discovery of the lack of conformance provisions that make these particular "application profiles" not "profiles" (3.4.1.3), i.e., the interoperability and compatibility relationships were not specified. These relationships need to be better specified if interoperability and compatibility are to be determined with respect to implementations of the base standard.